ABSTRACT

Title of Dissertation: ENHANCING VISUAL AND GESTURAL FIDELITY FOR EFFECTIVE VIRTUAL ENVIRONMENTS

Xiaoxu Meng
Doctor of Philosophy, 2020

Directed by: Professor Amitabh Varshney
Department of Computer Science

A challenge for the virtual reality (VR) industry is facing is that VR is not immersive enough to make people feel a genuine sense of presence: the low frame rate leads to dizziness and the lack of human body visualization limits the human-computer interaction. In this dissertation, I present our research on enhancing visual and gestural fidelity in the virtual environment.

First, I present a new foveated rendering technique: Kernel Foveated Rendering (KFR), which parameterizes foveated rendering by embedding polynomial kernel functions in log-polar space. This GPU-driven technique uses parameterized foveation that mimics the distribution of photoreceptors in the human retina. I present a two-pass kernel foveated rendering pipeline that maps well onto modern GPUs. I have carried out user studies to empirically identify the KFR parameters and have observed a $2.8\times - 3.2\times$ speedup in rendering on $4K$ displays.

Second, I explore the rendering acceleration through foveation for 4D light fields, which captures both the spatial and angular rays, thus enabling free-viewpoint

rendering and custom selection of the focal plane. I optimize the KFR algorithm by adjusting the weight of each slice in the light field, so that it automatically selects the optimal foveation parameters for different images according to the gaze position. I have validated our approach on the rendering of light fields by carrying out both quantitative experiments and user studies. Our method achieves speedups of $3.47\times - 7.28\times$ for different levels of foveation and different rendering resolutions.

Thirdly, I present a simple yet effective technique for further reducing the cost of foveated rendering by leveraging ocular dominance - the tendency of the human visual system to prefer scene perception from one eye over the other. Our new approach, eye-dominance-guided foveated rendering (EFR), renders the scene at a lower foveation level (with higher detail) for the dominant eye than the non-dominant eye. Compared with traditional foveated rendering, EFR can be expected to provide superior rendering performance while preserving the same level of perceived visual quality.

Finally, I present an approach to use an end-to-end convolutional neural network, which consists of a concatenation of an encoder and a decoder, to reconstruct a 3D model of a human hand from a single RGB image. Previous research work on hand mesh reconstruction suffers from the lack of training data. To train networks with full supervision, we fit a parametric hand model to 3D annotations, and we train the networks with the RGB image with the fitted parametric model as the supervision. Our approach leads to significantly improved quality compared to state-of-the-art hand mesh reconstruction techniques.

ENHANCING VISUAL AND GESTURAL FIDELITY FOR EFFECTIVE VIRTUAL ENVIRONMENTS

by

Xiaoxu Meng

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2020

Advisory Committee:
Professor Amitabh Varshney, Chair/Advisor
Professor Joseph F. JaJa
Professor Matthias Zwicker
Professor Roger Eastman
Professor Furong Huang

Acknowledgments

I owe my gratitude to all the people who have made this dissertation possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost I'd like to thank my advisor, Professor Amitabh Varshney for giving me an invaluable opportunity to work on the challenging and interesting projects over the past five years. He has always made himself available for help and advice and there has never been an occasion when I've knocked on his door and he hasn't given me time. It has been a pleasure to work with and learn from such an extraordinary individual.

Thanks are due to Professor Matthias Zwicker, Professor Joseph F. JaJa, Professor Furong Huang, and Professor Roger Eastman for agreeing to serve on my dissertation committee and for sparing his/her invaluable time.

I would also like to thank Dr. Michael Hemmer and Dr. Kun He for their supervision during my internship at Google and Facebook, respectively.

My colleagues at the Graphics and Visual Informatics Laboratory (GVIL) have enriched my graduate life in many ways and deserve a special mention. Dr. Ruofei Du inspired my research and helped me in solving technical problems. My interactions with Dr. Hsueh-Chien Cheng, Dr. Eric Krokos, Xuetong Sun, Tara Larrue, Shuo Li, Somay Jain, Mukul Agarwal, Alexander Rowden, Susmija Reddy Jabbireddy and David Li have been very fruitful.

I would like to acknowledge my classmates and friends, including Yuexi Chen, Sheng Cheng, Hui Ding, Fenfei Guo, Yue Jiang, Gregory Kramida, Shang Li, Yuntao

Liu, Zhenyu Lin, Zijie Lin, Tiantao Lu, Nitin J. Sanket, Sayyed Sina Miran, Jiahao Su, Guowei Sun, Manasij Venkatesh, Harry Wandersman, Qian Wu, Pengcheng Xu, Xi Yi and Daiwei Zhu.

I would also like to acknowledge help and support from the staff members from UMIACS and the CS department. Tom Ventsias, Barbara Brawn-Cinani, Sida Li, Eric Lee, Jonathan Heagerty, Vivian Lu, and Tom Hurst's technical help and encouragement are highly appreciated.

I owe my deepest thanks to my family – my mother and father who have always stood by me and guided me through my career, and have pulled me through against impossible odds at times. Words cannot express the gratitude I owe them. I would also like to thank Yunchuan Li, Hua Luo, and Hong Wei who are like family members to me.

I would like to acknowledge financial support from UMIACS for all the projects discussed herein.

It is impossible to remember all, and I apologize to those I've inadvertently left out.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Overview

Rendering speed and transmission bandwidth are two critical constraints in realizing effective and distributed virtual reality [22]. Human vision spans a field of view of $135^\circ \times 160^\circ$, but the highest-resolution foveal vision covers only the central $1.5^\circ \times 2^\circ$ [6]. Patney *et al.* [8] have estimated that in modern virtual reality head-mounted displays (HMD) only 4% of the pixels are mapped onto the fovea. Foveated rendering [6, 13, 23] aims to improve the rendering efficiency while maintaining visual quality by leveraging the capabilities and the limitations of the human visual system. Equipped with an eye-tracker, a foveated rendering system presents the foveal vision with full-resolution rendering and the peripheral vision with low-resolution rendering. This allows one to improve the overall rendering performance while maintaining high visual fidelity. Therefore, foveated rendering techniques that allocate more computational resources for foveal pixels and fewer resources elsewhere can dramatically speed up rendering [24] for large displays, especially for virtual and augmented reality headsets equipped with eye trackers.

There is a large research literature that documents the falloff of accuracy in visual periphery. Most of the previous research on foveated perception has been

surveyed in [25]. Recent research has also addressed the issue of perception time for depicting information in the far peripheral field [26]. A commonly used model is the *linear acuity model*, which measures the minimum angle of resolution (MAR). A linear model matches both anatomical data and is applicable for many low-level vision tasks [25]. However, the model only works for the "central" vision (with angular radius $\leq 8°$), after which MAR rises more steeply [6]. In the periphery, receptors become increasingly sparse relative to the eyes' optical system Nyquist limit. Another model is the *log acuity model*. It has been found that the excitation of the cortex can be approximated by a log-polar mapping of the eye's retinal image [27]. The calculation of this model is cheap and fast, thus being used in many practical applications such as computer vision, robotics, and other fields. Curcio [1,2] proposed the mixed acuity model. As shown in Figure 1.1, the Ganglion cell (orange) density tends to match the photoreceptor density in the fovea (left), but many photoreceptors map to the same Ganglion cell away from the fovea. 'Nasal', 'Temp', 'Sup' and 'Inf' indicate the four directions (nasal, temporal, superior, inferior) away from the fovea.

Most of the foveated rendering algorithms are inspired by the acuity models mentioned above. However, it is not easy to quantify the pixel density rate by using the models mentioned above. Previous research has addressed the issue of adjusting visual acuity by conducting user studies with eye tracking technologies.

In this dissertation, I first present kernel foveated rendering for rendering 3D meshes and ray-traced scenes. I parameterize the foveation of rendering by embedding polynomial kernel functions in the classic log-polar mapping [28,29]. This allows us to alter both the sampling density and distribution, and match them to

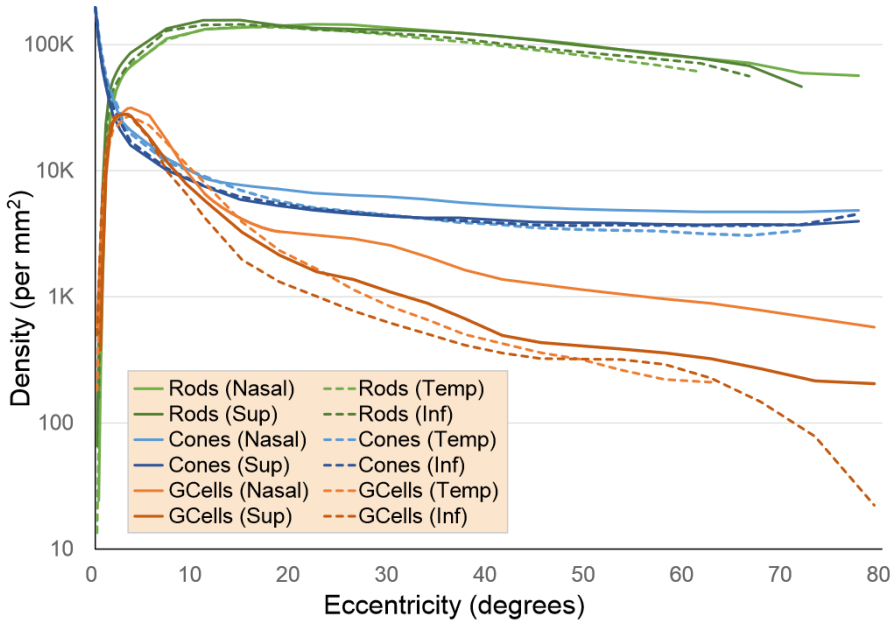


Figure 1.1: Spatial distribution of various retinal components, using data from [1] and [2]. Ganglion cell (orange) density tends to match photoreceptor density in the fovea (left), but away from the fovea many photoreceptors map to the same Ganglion cell. 'Nasal', 'Temp', 'Sup' and 'Inf' indicate the four directions (nasal, temporal, superior, inferior) away from the fovea.

human perception in virtual reality HMDs.

Second, I optimize 3D-KFR by adjusting the weight of each slice in the light fields, so that it automatically selects the optimal foveation parameters for different images according to the gaze position and achieves higher speedup. In this way, 3D-KFR further accelerates the rendering process of high-resolution light fields while preserving the perceptually accurate foveal detail.

Third, I present a simple yet effective technique for further reducing the cost of foveated rendering by leveraging ocular dominance - the tendency of the human visual system to prefer scene perception from one eye over the other. Our approach, eye-dominance-guided foveated rendering (EFR), renders the scene with better detail for the dominant-eye than the non-dominant-eye. Compared with traditional foveated rendering, EFR provides superior rendering efficiency while preserving the same level of perceived visual quality.

Finally, I present an end-to-end convolutional autoencoder to reconstruct a 3D human hand from a single RGB image. To train networks with full supervision, we fit a parametric hand model to 3D annotations, and we train the networks with the RGB image with the fitted parametric model as the supervision. Our approach leads to significantly improved quality compared to state-of-the-art hand mesh reconstruction techniques.

1.2 Kernel Foveated Rendering for 3D Graphics

In Chapter 2, I present the kernel foveated rendering (KFR) for 3D graphics [23], a foveated rendering system with smoothly changing resolution from fovea to the periphery as shown in Figure 1.2.

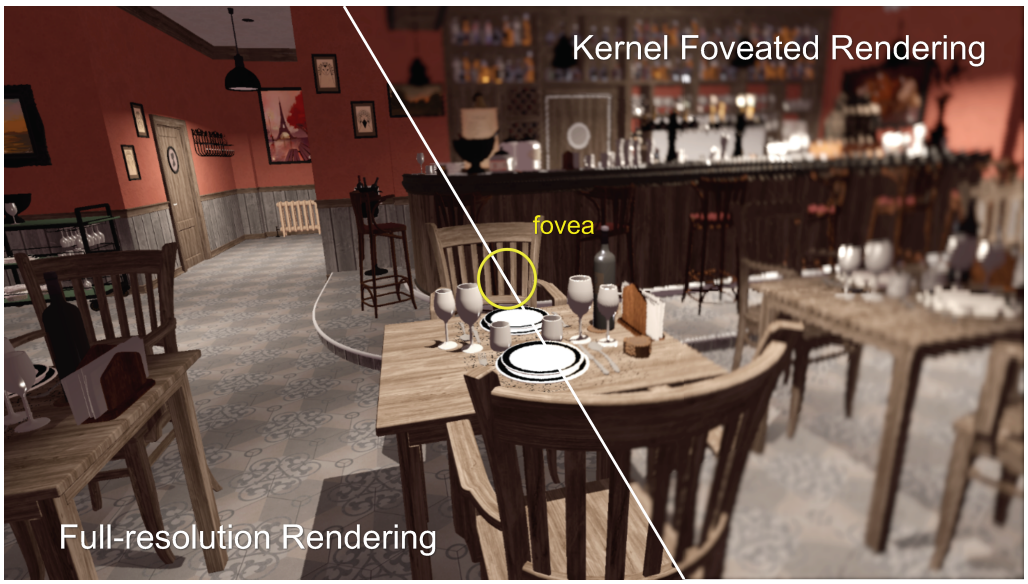


Figure 1.2: The comparison of the full-resolution rendering (left) and kernel foveated rendering (right). Kernel foveated rendering system mimics the distribution of photoreceptors in the human retina and generates foveated rendering with smoothly changing resolution.

In the KFR rendering system, I parameterize foveated rendering by embedding polynomial kernel functions in the classic log-polar mapping. The GPU-driven technique uses closed-form, parameterized foveation that mimics the distribution of photoreceptors in the human retina. The pipeline of kernel foveated rendering contains two passes as shown in Figure 1.3.

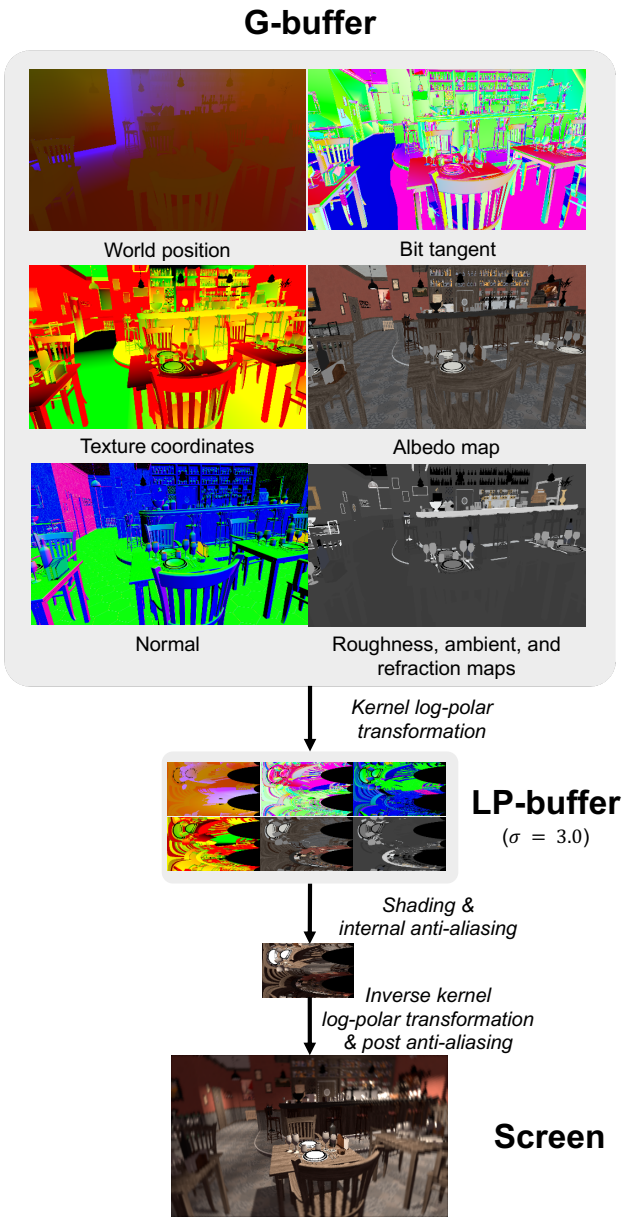


Figure 1.3: An overview of the kernel foveated rendering pipeline. I transform the necessary parameters and textures in the G-buffer from Cartesian coordinates to log-polar coordinates, compute lighting in the log-polar (LP) buffer and perform internal anti-aliasing. Next, I apply the inverse transformation to recover the frame buffer in Cartesian coordinates and employ post anti-aliasing to reduce the foveation artifacts.

In the first pass, I compute the kernel log-polar transformation of the necessary parameters and textures in the G-buffer from Cartesian coordinates to log-polar coordinates and store the transformation in a reduced-resolution log-polar (LP) buffer. Then I compute shading in the LP buffer; the shading cost is greatly reduced because the lighting calculations at the reduced-resolution are lower. Due to the low-resolution of the LP-buffer, there may be artifacts in the peripheral regions after the inverse transformation. Therefore, I add a denoising stage in the log-polar space. To reduce artifacts in the peripheral regions, I use a Gaussian filter with a 3×3 kernel for the right part of the rendering(corresponding to the peripheral regions) in the LP-buffer.

In the second pass, I carry out the inverse-log-polar transformation to recover the rendered image in log-polar coordinates to the Cartesian coordinates for display. I also perform spacial and temporal anti-aliasing for the full-resolution image. To empirically establish the most suitable foveation parameter values, I have carried out pilot and formal user studies. To achieve visually acceptable results for foveated rendering, I use a threshold of 80% responses considering foveated rendering to be visually indistinguishable from full-resolution rendering. I therefore choose $\sigma = 1.8$ and $\alpha = 4$ as the desired parameters for the interactive rendering evaluation. With the desired parameters, I observe a $2.8\times$ – $3.2\times$ speedup in rendering on $4K$ UHD ($2160p$) displays with minimal perceptual loss of detail. The relevance of eye-tracking-guided kernel foveated rendering can only increase as the anticipated growth of display resolution makes it ever more difficult to resolve the mutually conflicting goals of interactive rendering and perceptual realism.

1.3 3D-Kernel Foveated Rendering for Light Fields

Light fields capture both the spatial and angular rays, thus enabling free-viewpoint rendering and custom selection of the focal plane. Scientists can interactively explore microscopic light fields of organs, microbes, and neurons using virtual reality headsets. However, rendering high-resolution light fields at interactive frame rates requires a very high rate of texture sampling, which is a challenge as the resolutions of light fields and displays continue to increase. In Chapter 3, I present 3D-kernel foveated rendering for light fields [30], a foveation system for light field as shown in Figure 1.4.

I have developed a perceptual model for foveated light fields by extending the KFR for the rendering of 3D meshes: since the foveation level of a pixel is affected by the distance to the center camera, the foveation parameter can be different for different slices in a light-field image array. We optimize the KFR algorithm into 3D-KFR by adjusting the weight of each slice in the light field, so that it is able to automatically select the optimal foveation parameters for different images according to the gaze position, thereby achieving greater speedup with minimal perceptual loss as shown in Figure 1.5.

3D-KFR coupled with eye-tracking can accelerate the rendering of 4D depth-cued light fields dramatically. On datasets of high-resolution microscopic light fields, I observe 3.47×-7.28× speedup in light field rendering with minimal perceptual loss of detail. I envision that 3D-KFR will reconcile the mutually conflicting goals of visual fidelity and rendering speed for interactive visualization of light fields.

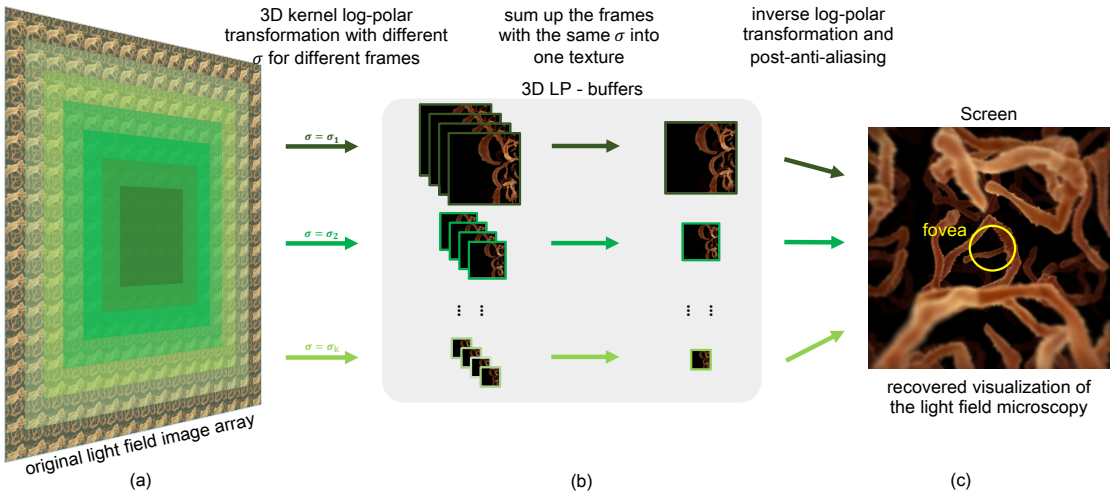


Figure 1.4: The pipeline of the foveated light field. Part (a) represents the light field image array, the region with dark-green mask represents the *foveal region* that the local center camera position of the frames are around fovea. The *peripheral region* masked by light-green are the regions that the local center camera position of the frames is far from the fovea. I apply kernel log-polar transformation for each image with different σ (σ is determined by the gaze position) to get the image sub-arrays as shown in the left part of (b). Then I average the image sub-arrays to get textures as shown in the right part of (b). Finally, I apply the inverse log-polar transformation for each sub-array, calculate the weighted sum of pixel values and perform anti-aliasing to get the final image displayed on-screen as shown in part (c).

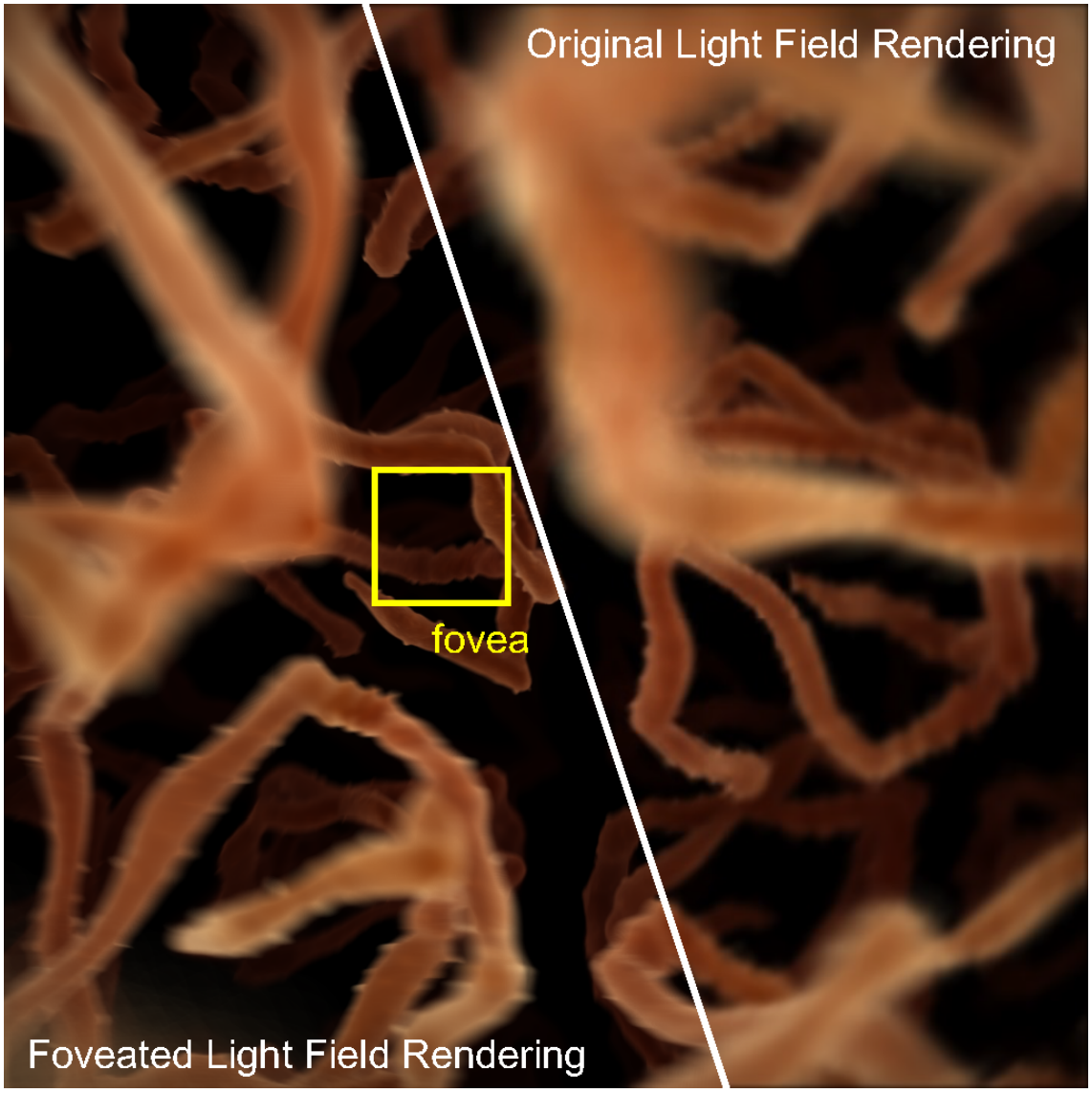


Figure 1.5: The comparison of the original full-resolution light field rendering (right) and foveated light field rendering (left). We optimize the KFR algorithm into 3D-KFR by adjusting the weight of each slice in the light field, so that it is able to automatically select the optimal foveation parameters for different images according to the gaze position, thereby achieving greater speedup minimal perceptual loss.

1.4 Eye-dominance-guided Foveated Rendering

In Chapter 4, I introduce eye-dominance-guided foveated rendering [31] as shown in Figure 1.6.

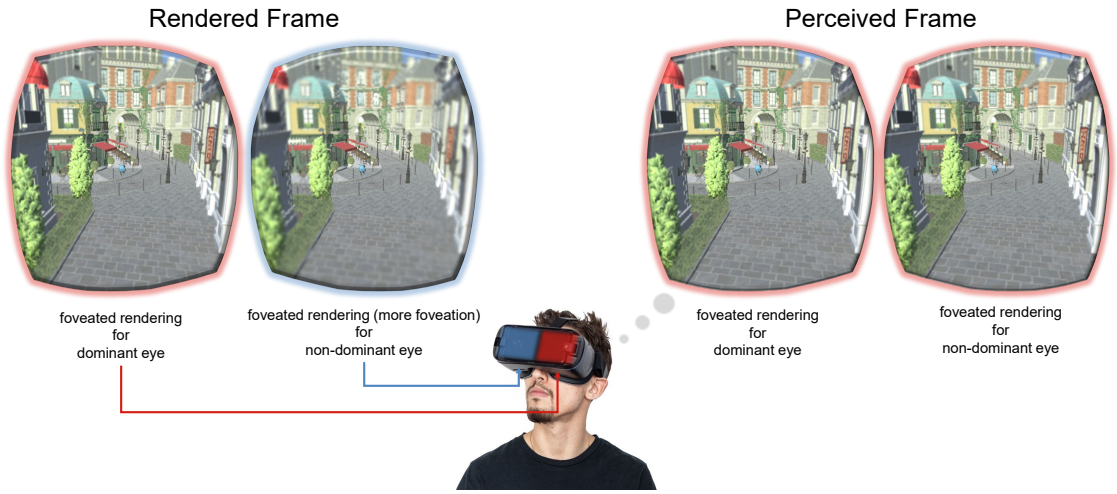


Figure 1.6: Our pipeline renders the frames displayed in the dominant eye at a lower foveation level (with higher detail), and renders the frames for the non-dominant eye at a higher foveation level. This improves rendering performance over traditional foveated rendering with minimal perceptual difference.

Here, I present a simple yet effective technique for further reducing the cost of foveated rendering by leveraging *ocular dominance* – the tendency of the human visual system to prefer scene perception from one eye over the other. I present the technique of eye-dominance-guided foveated rendering (EFR), which leverages ocular dominance property of the human visual system. I render the scene for the dominant eye at the normal foveation level and render the scene for the non-dominant eye at

a higher foveation level. This formulation allows us to save more in the rendering budget for the non-dominant eye.

I have validated our approach by carrying out quantitative experiments and user studies. I designed two user tests to establish the most suitable foveation parameter values for the dominant eye and the parameter for the non-dominant eye. I have implemented the eye-dominance-guided foveated rendering pipeline on a GPU, and achieve up to 1.47× speedup compared with the original foveated rendering at a resolution of 1280×1440 per eye with minimal perceptual loss of detail. The technique of eye-dominance-guided foveated rendering can be easily integrated into the current rasterization rendering pipeline for head-mounted displays.

1.5 Hand Mesh Reconstruction from a RGB Image

Accurate reconstruction of 3D human hands from monocular RGB images is a challenging task. Hand estimation benefits a broad range of applications, such as human-computer interaction and virtual and augmented reality. The goal of this research is to use an end-to-end deep neural network to reconstruct the 3D model of a human hand from a single RGB image as shown in Figure 1.7. In Chapter 5, I present a network architecture as a concatenation of an encoder and a decoder. Given a single RGB image of a hand, the encoder predicts the feature vector, from which the decoder decodes a 3D hand model. To train networks with full supervision, we fit a parametric hand model to 3D annotations, and we train the networks with the RGB image with the fitted parametric model as the supervision. Our approach leads to

significantly improved quality compared to state-of-the-art hand mesh reconstruction techniques.

We envision that the proposed approach could be widely used in the human-object-interaction by facilitating the interaction between users and virtual objects and bring virtual reality users more immersive experience by the visualization of their hand model.

Figure 1.7: Example of the estimation of the hand pose and shape. The $Fscore@10mm$ indicates the accuracy of estimation (higher is better).

Chapter 2: Kernel Foveated Rendering for 3D Graphics

2.1 Overview

Araujo and Dias [28] use a log-polar mapping to approximate the excitation of the cortex in the human visual system. The classic log-polar transformation has been used for foveating 2D images on the GPU [29]. However, to the best of my knowledge, direct use of the log-polar mapping for 3D graphics has not yet been attempted on GPUs.

In this chapter, I present a kernel foveated rendering pipeline for modern GPUs that parameterizes foveated rendering by embedding polynomial kernel functions in the classic log-polar mapping. This allows us to easily vary the sampling density and distribution, and match them to human perception in virtual reality HMDs. In contrast to adaptive sampling in Cartesian coordinates, which requires a complex interpolation process [13] and the classic three-pass foveated rendering pipeline [6], KFR just needs a two-pass algorithm. In the first pass, I carry out the kernel log-polar transformation and render to a reduced-resolution framebuffer using deferred shading [32,33]. In the second pass, I apply the inverse kernel log-polar transformation to the reduced-resolution framebuffer to map the final foveated rendering to the full-resolution display.

I have built several foveated renderings with varying sampling density and distribution and evaluate them via pilot and final user studies. I have found the optimal parameters with minimal perceptual errors that correspond to the distribution of photoreceptors in the retina. This algorithm is designed to achieve a high frame rate by shading fewer pixels in the peripheral vision. Finally, I show results of validating my approach on 3D rendering of textured meshes as well as ray-marching scenes.

The KFR pipeline is broadly applicable for eye-tracking devices, and efficiently testing, or previewing real-time rendering results with global lighting and physically based rendering.

In summary, my contributions include:

1. designing the kernel log-polar mapping algorithm to enable a parameterized trade-off of visual quality and rendering speed for foveated rendering,

2. conducting user studies to identify the kernel foveated rendering parameters governing the sampling distribution and density to maximize perceptual realism and minimize computation,

3. mapping kernel foveated rendering onto the GPU to achieve speedups of $2.8X$ for textured 3D meshes and $3.2X$ for ray-casting scenes for 3840×2160 displays with the minimal perceived loss of detail.

2.2 Related Work

In this section, I will review the development of foveated rendering in images, videos and 3D renderings and give an overview of the eye-tracking technology.

2.2.1 Foveated Images and Videos

The last few decades have seen significant advances in foveated rendering for 2D images and videos.

Burt [34] has generated foveated images with multi-resolution Gaussian pyramids. He takes advantage of a coarse-to-fine scheme to adaptively select the critical information for constructing the foveated image. Kortum and Kortum *et al.* [3] have developed one of the earliest eye-tracking-based foveated imaging systems with space-variant degradation. The structure of their image foveation system is shown in Figure 2.1. Using 256×256 8-bit gray-scale images, they have achieved bandwidth reduction of up to 94.7% with minimal perceptual artifacts. Other image foveation techniques include embedded zero-tree wavelets [35], set partitioning in hierarchical trees [36], wavelet-based image foveation [37], embedded foveation image coding [38], and gigapixel displays [39].

Video foveation has also been explored [40–42]. The filter bank method is used for video preprocessing before using standard video compression algorithms (*e.g. MPEG and H.26x*) [41, 43, 44]. Foveation filtering has been implemented with the quantization processes in standard MPEG and H.26x compression [45, 46].

Video foveation coupled with eye tracking could reduce overall system latency,

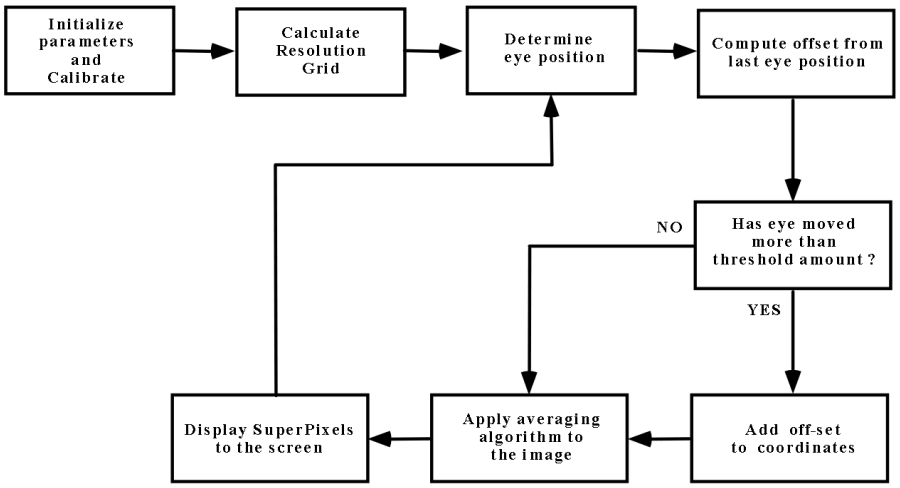


Figure 2.1: Kortum [3] have developed one of the earliest eye-tracking-based foveated imaging systems with space-variant degradation.

including network latency, processing latency, and display latency. Lungaro *et al.* [4,5] propose to use a tile-based foveated rendering algorithm to reduce the overall bandwidth requirements for streaming 360° videos. An overview of video foveation is shown in Figure 2.2. They quantified the bandwidth savings achievable by the proposed approach and characterize the relationships between Quality of Experience (QoE) and network latency. The results showed that up to 83% less bandwidth is required to deliver high QoE levels to the users, as compared to conventional solutions.

As shown in Figure 2.3, Kaplanyan *et al.* present *DeepFovea* [47], which uses a generative adversarial network [48] to reconstruct the given sparsely foveated image by considering the closest match on a learned manifold of natural videos. Given a history of frames and the corresponding gaze points that the user sees till a given timestamp, along with the sparsely constructed image at that time frame, the deep

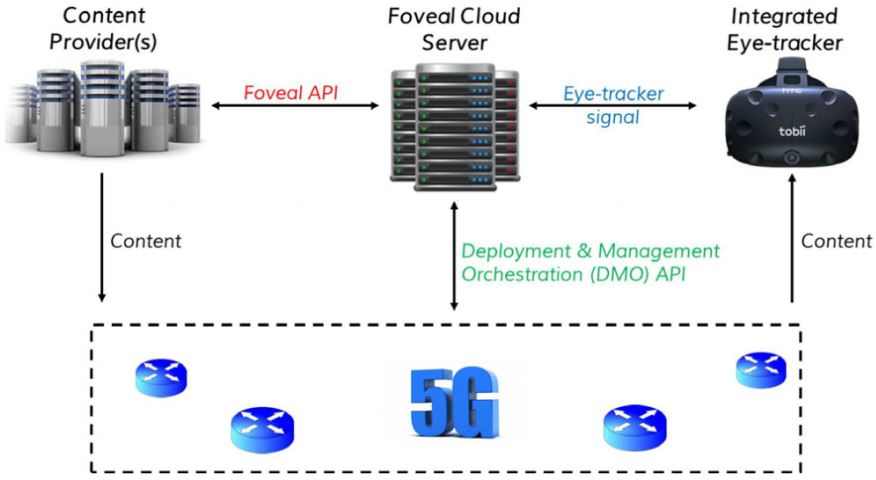


Figure 2.2: Lungaro *et al.* [4, 5] propose to use a tile-based foveated rendering algorithm to reduce the overall bandwidth requirements for streaming 360° videos.

neural network reconstructs the original frame by inpainting and in-hallucinating peripheral details while maintaining high acuity at the gaze point.

While previous work in foveation for images and videos provides strong foundations, most of these methods cannot be easily generalized for interactive 3D graphics rendering on modern GPUs. A notable exception is the work by Antonelli *et al.* [29], which uses log-polar mapping to speed-up 2D image rendering on modern GPUs. However, their approach does not directly work with 3D graphics primitives and does not use kernel functions.

2.2.2 Foveated 3D Graphics

Weier *et al.* [49] have reviewed several approaches for foveated rendering including mesh simplification in the areas of lower acuity [50–52]. However, these days shading has often been found to dominate the cost for rendering sophisticated

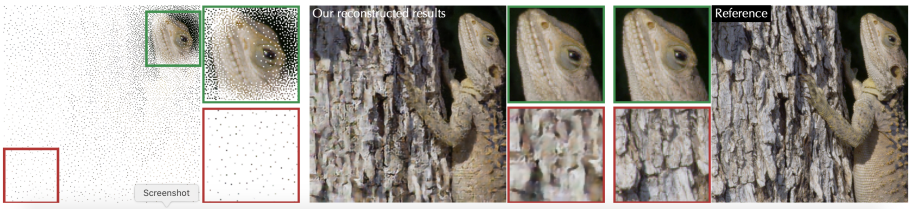


Figure 2.3: Foveated reconstruction with DeepFovea. Left to right: (1) sparse foveated video frame (gaze in the upper right) with 10% of pixels; (2) a frame reconstructed from it with our reconstruction method; and (3) full resolution reference. Our method in-hallucinates missing details based on the spatial and temporal context provided by the stream of sparse pixels. It achieves 14× compression on RGB video with no significant degradation in perceived quality. Zoom-ins show the 0 foveal and 30 periphery regions with different pixel densities. Note it is impossible to assess peripheral quality with your foveal vision.

scenes on modern graphics pipelines [7, 11].

Ragan-Kelley *et al.* [53] use decoupled sampling for stochastic super-sampling of motion and defocus blur at a reduced shading cost. Guenter *et al.* [6] present a three-pass pipeline for foveated 3D rendering by using three eccentricity layers around the tracked gaze point. As shown in Figure 2.4, the innermost layer is rendered at the highest resolution (native display), while the successively outer peripheral layers are rendered with progressively lower resolution and coarser level of detail (LOD). They interpolate and blend between the layers and use frame jitter and temporal re-projection to reduce spatial and temporal artifacts. However, this approach renders the scene three times, which requires lots of rendering resources.

Vaidyanathan *et al.* [7] present a novel approach using a generalization of

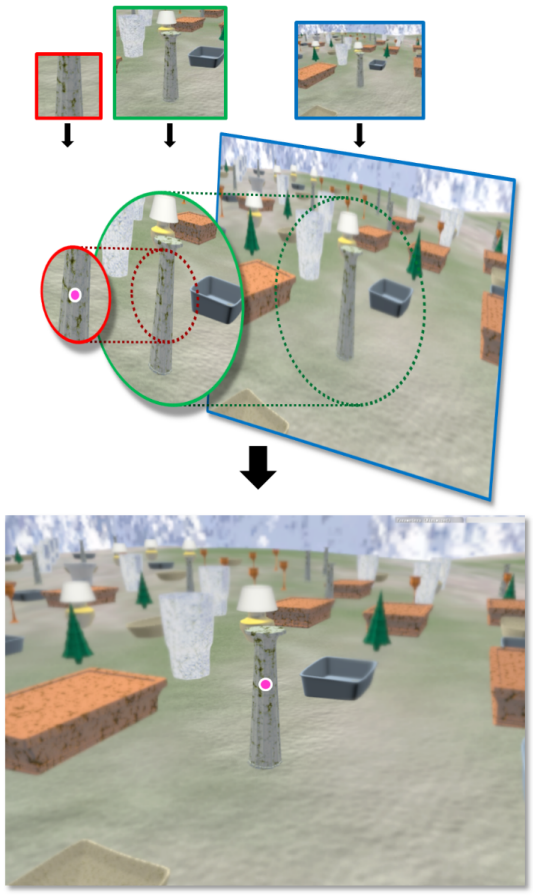


Figure 2.4: Guenter *et al.* [6] render three eccentricity layers (red border = inner layer, green = middle layer, blue = outer layer) around the tracked gaze point (pink dot), shown at their correct relative sizes in the top row. These are interpolated to native display resolution and smoothly composited to yield the final image at the bottom. Foveated rendering greatly reduces the number of pixels shaded and overall graphics computation.

multi-sample anti-aliasing (MSAA). They perform foveated rendering by sampling coarse pixels (2×2 pixels and 4×4 pixels) in the peripheral regions as shown in Figure 2.5. This approach targets small-form-factor devices with high resolution, such as phones and tablets rather than HMDs. It therefore presents two challenges for HMDs: the effective pixel size in current HMDs is too large for MSAA, and gaze-dependent motions exaggerate the artifacts.

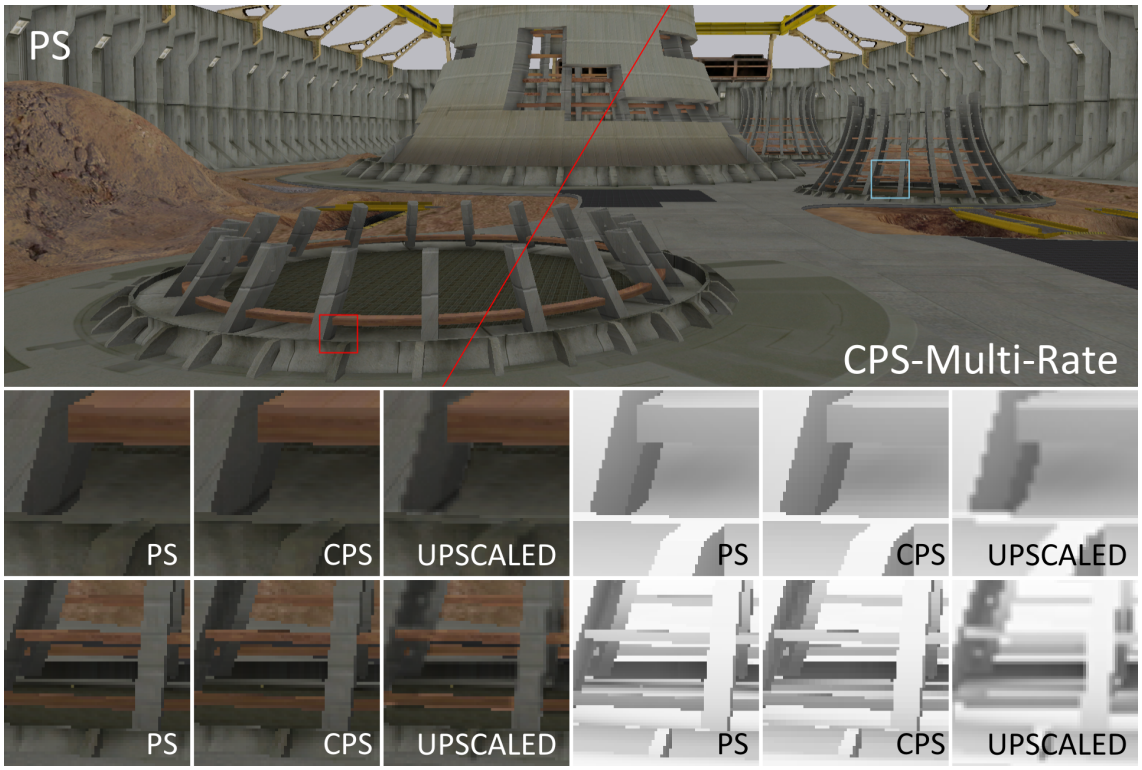


Figure 2.5: Vaidyanathan *et al.* [7] perform foveated rendering by sampling coarse pixels (2×2 pixels and 4×4 pixels) in the peripheral regions.

Patney *et al.* [8, 9] address temporal artifacts in foveated rendering by using pre-filters and temporal anti-aliasing. Because human eyes are sensitive to edges, they add contrast preservation for the foveated image, which greatly enhances the

image quality by reducing the tunneling effect as shown in Figure 2.6. They tested the foveated rendering effect on both Desktop and VR headset.

Figure 2.6: Patney *et al.* [8,9] perform foveated rendering by sampling coarse pixels and address temporal artifacts in foveated rendering by using pre-filters and temporal anti-aliasing.

Clarberg *et al.* [10] propose a modification to the current hardware architecture, which enables flexible control of shading rates and automatic shading reuse between triangles in tessellated primitives as shown in Figure 2.7.

He *et al.* [11] introduce multi-rate GPU shading to support more shading samples near regions of specular highlights, shadows, edges, and motion blur regions, helping achieve a $3X$ to $5X$ speedup as shown in Figure 2.8. However, this implementation of multi-rate shading requires an extension of the graphics pipeline, which

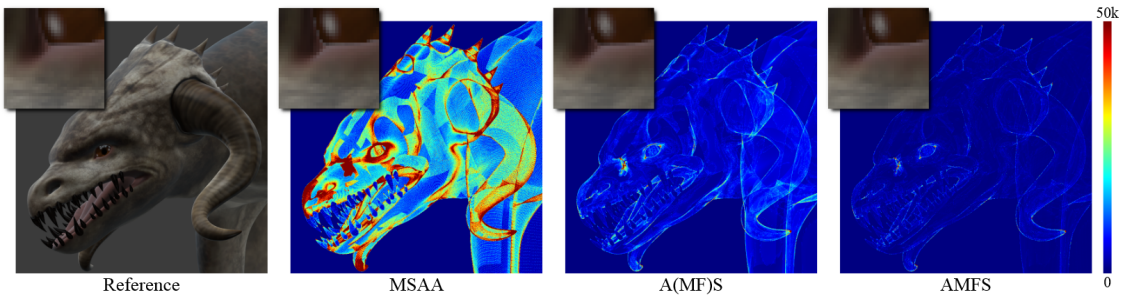


Figure 2.7: Clarberg *et al.* [10] have proposed an approach in which pixel shading is tied to the coarse input patches and reused between triangles, effectively decoupling the shading cost from the tessellation level, as shown in this example.

is not available on commodity graphics hardware.

Swafford *et al.* [12] implement four foveated renderers as shown in Figure 2.9. The first method reduces the effective rendered pixel density of the peripheral region while maintaining the base density of the foveal window, as shown in Figure 2.9 (a). The second varies per-pixel depth-buffer samples in the fovea and periphery for screen-space ambient occlusion. Although a very low number of per-pixel samples can cause banding, they expect these differences to go unnoticed in the periphery due to the loss of visual acuity and contrast sensitivity, as shown in Figure 2.9 (b). The third method normally casts rays to geometry and detects intersections with a given number of depth layers, represented as a series of RGBA textures mapped on the geometry, then it varies the per-pixel ray-casting steps across the field of view, as shown in Figure 2.9 (c). The final method implements a terrain renderer using GPU-level tessellation for the fovea. In order to determine the appropriate level of

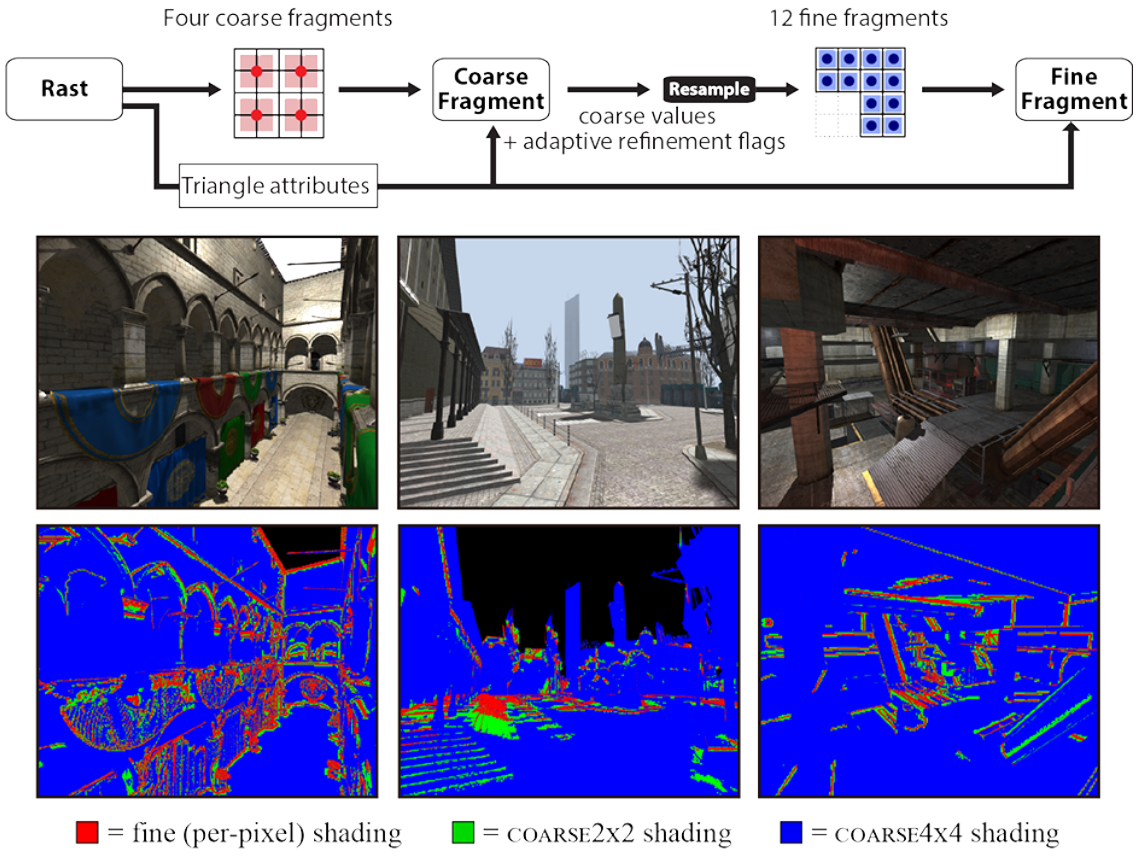


Figure 2.8: He *et al.* [11] introduce multi-rate GPU shading to support more shading samples near regions of specular highlights, shadows, edges, and motion blur regions, helping achieve a $3X$ to $5X$ speedup.

tessellation, we project the foveal window from screen coordinates into the scene. If a tile falls within either the foveal or peripheral field of view, the level of tessellation is set statically to the appropriate level. If the tile falls between the two regions (on the blending border) the level of tessellation is linearly interpolated between the two levels, as shown in Figure 2.9 (d).

Stengel *et al.* [13] use adaptive sampling from fovea to peripheral regions in a gaze-contingent rendering pipeline as shown in Figure 2.10. To compensate for the missing pixels caused by sparsely distributed shading samples on the periphery, they use pull-push [54] interpolation to create the full foveated image. This strategy achieves a reduction of render time of 25.4% (with speedup of $1.3X$) and reduction of shading time of 41% (with speedup of $1.7X$).

As shown in Figure 2.11, Tursun [14] propose luminance-contrast-aware foveated rendring, which demonstrates that the computational savings of foveated rendering can be significantly improved if local luminance contrast of the image is analyzed. They first study the resolution requirements at different eccentricities as a function of luminance patterns. They later use this information to derive a low-cost predictor of the foveated rendering parameters. Its main feature is the ability to predict the parameters using only a low-resolution version of the current frame, even though the prediction holds for high-resolution rendering.

Besides virtual reality, foveated rendering is also desirable for augmented reality (AR) [55]. As shown in Figure 2.12, the AR display combines a traveling micro-display relayed off a concave half-mirror magnifier for the high-resolution foveal region, with a wide field-of-view peripheral display using a projector-based Maxwellian-view display

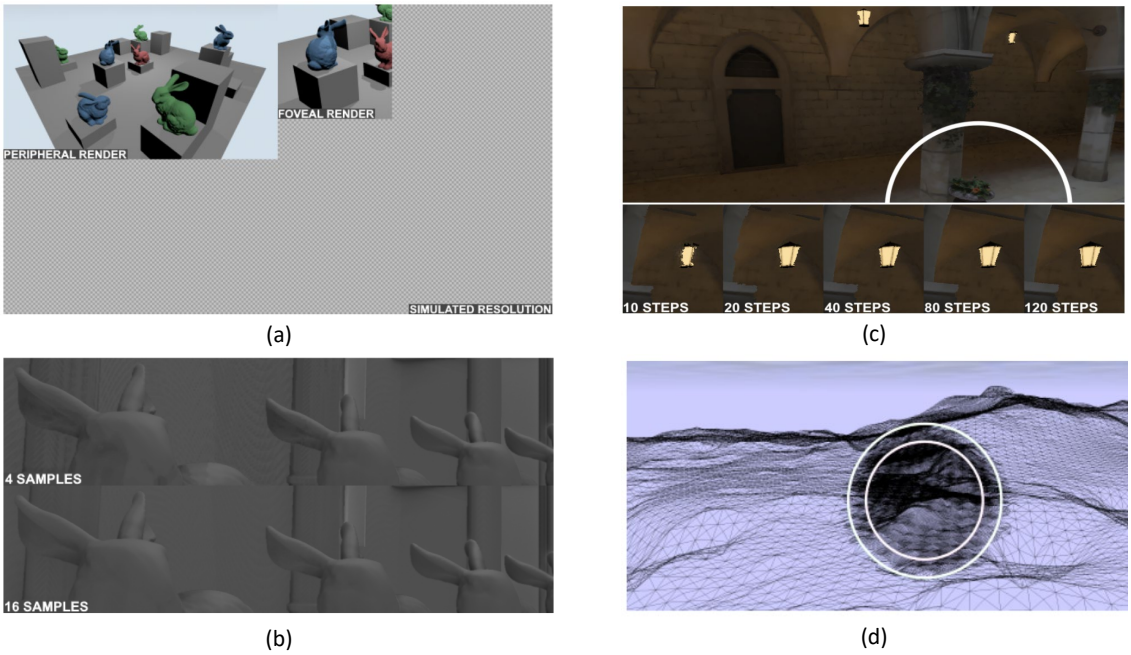


Figure 2.9: Swafford *et al.* [12] implement four foveated renderers as described in the text of the figure. (a) is the annotated view of a foveated render with moderate settings pre-composition. The checkerboard area represents the proportion of pixels saved for the targeted simulated resolution; (b) is the strips from two foveated renders with the same fixation point (bottom-right) but different peripheral sampling levels. Region transition is handled smoothly, but at four samples there are noticeable artifacts in the peripheral region, such as banding; (c) Top: Sample frame from our ray-casting method with 120 per-pixel steps in the foveal region (within circle) and 10 per-pixel steps in the peripheral region (outwith circle). Bottom: Close-up of right lamp showing artifacts across different quality levels; (d) is the Wireframe view of our foveated tessellation method. The inner circle is the foveal region, between circles is the inter-regional blending, and outside the circles is the peripheral region.

Figure 2.10: Stengel *et al.* [13] use adaptive sampling from fovea to peripheral regions in a gaze-contingent rendering pipeline and compensate for the missing pixels by pull-push interpolation.

Figure 2.11: Tursun [14] propose luminance-contrast-aware foveated rendring, which demonstrates that the computational savings of foveated rendering can be significantly improved if local luminance contrast of the image is analyzed.

whose nodal point is translated to follow the viewer's pupil during eye movements using a traveling holographic optical element.

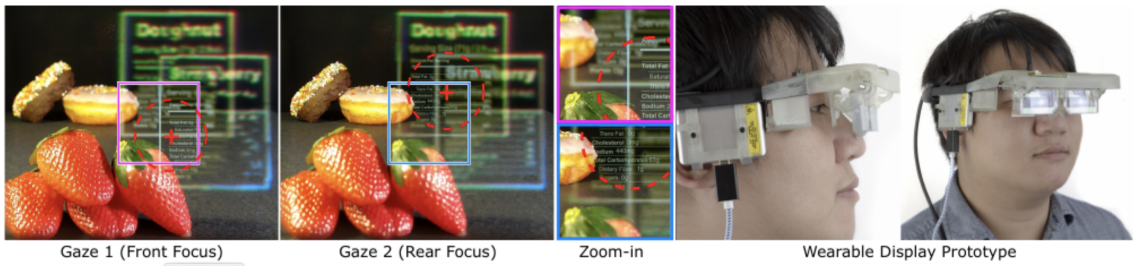


Figure 2.12: Display results from our Foveated AR prototype. By tracking the user's gaze direction (red cross), the system dynamically provides high-resolution inset images to the foveal region and low-resolution large-FOV images to the periphery. The system supports accommodation cues; the magenta and blue zoom-in panels show optical defocus of real objects together with foveated display of correctly defocus-blurred synthetic objects. Red dashed discs highlight the foveal vs peripheral display regions. A monocular wearable prototype (functional but manually actuated) illustrates the compact optical path.

Recently, deferred shading has been used for antialiasing foveated rendering. Karis [56] optimizes temporal anti-aliasing for deferred shading, which uses samples over multiple frames to reduce flickering. Crassin *et al.* [57] reduce aliasing by pre-filtering sub-pixel geometric detail in the G-buffer for deferred shading. Chajdas *et al.* [58]'s subpixel anti-aliasing operates as a post-process on a rendered image with super-resolution depth and normal buffers. It targets deferred shading renderers that cannot use MSAA.

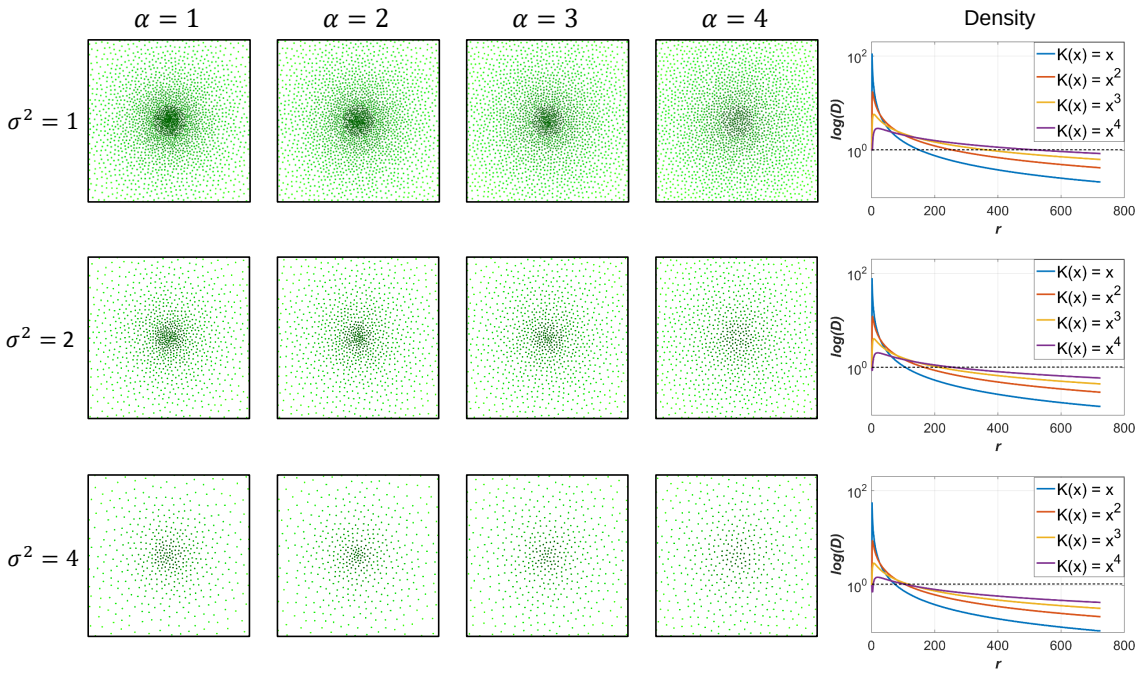


Figure 2.13: The relationship among σ^2, $\mathbf{K}(x) = x^\alpha$, and the sampling rate. The number of samples in each image is proportional to σ^2. I use a variant of the PixelPie algorithm [15] to generate the Poisson samples shown.

In this dissertation, I present a simple two-pass foveated rendering pipeline that maps well onto modern GPUs. Kernel foveated rendering (KFR) provides gradually changing resolution and achieves $2.8X - 3.2X$ speedup with little perceptual loss.

2.3 Proposed Algorithm

Overall, my algorithm applies the kernel log-polar transformation for rasterization in a reduced-resolution log-polar buffer (LP-buffer), carries out shading within the LP-buffer, and then uses the inverse kernel log-polar transformation to render

on the full resolution display. This is shown in Figure 1.3.

In the classic log-polar transformation [29], given a $W \times H$ pixel display screen, and an LP-buffer of $w \times h$ pixels, the screen-space pixel (x, y) in Cartesian coordinates is transformed to (u, v) in the log-polar coordinates according to Equation 2.1,

$$\begin{aligned} u &= \frac{\log\|x', y'\|_2}{L} \cdot w \\ v &= \frac{\arctan\left(\frac{y'}{x'}\right)}{2\pi} \cdot h + \mathbf{1}\left[y' < 0\right] \cdot h \end{aligned} \tag{2.1}$$

where, (x', y') represent (x, y) with respect to the center of the screen as origin, L is the log-distance from the center to the corner of the screen, and $\mathbf{1}[\cdot]$ is the indicator function,

$$x' = x - \frac{W}{2}, \quad y' = y - \frac{H}{2}, \quad L = \log\left(\left\|\frac{W}{2}, \frac{H}{2}\right\|_2\right) \tag{2.2}$$

$$\mathbf{1}\left[y' < 0\right] = \begin{cases} 1 & , y' < 0 \\ 0 & , y' \geq 0 \end{cases} \tag{2.3}$$

Notice how the central dark green area in Figure 2.14 (a) is mapped to a relatively large region in the left part of the log-polar coordinates in Figure 2.14 (b), while the peripheral regions of Figure 2.14 (a) are mapped to a relatively small part of Figure 2.14 (b).

In the inverse log-polar transformation, a pixel with log-polar coordinates (u, v) is transformed back to (x'', y'') in Cartesian coordinates. Let

$$A = \frac{L}{w}, \quad B = \frac{2\pi}{h}, \tag{2.4}$$

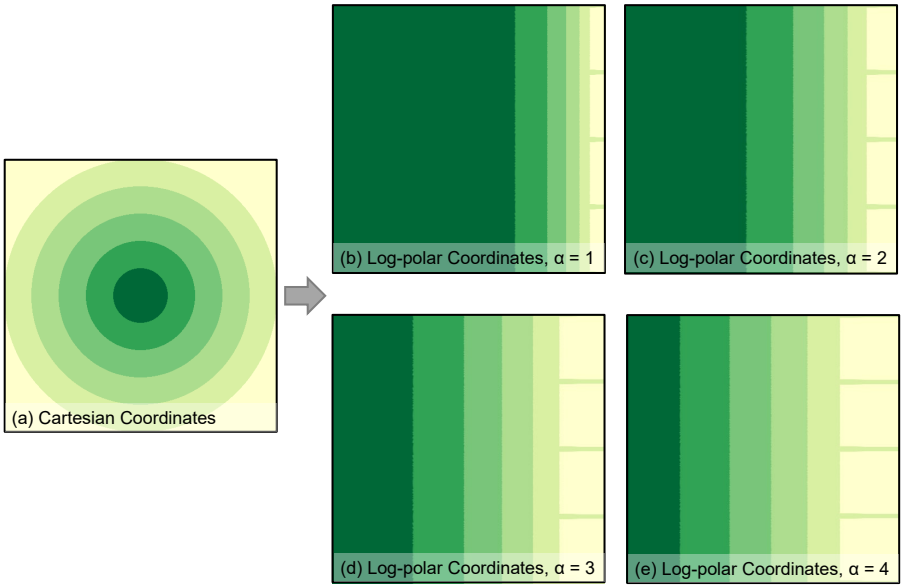


Figure 2.14: Transformation from Cartesian coordinates to log-polar coordinates with kernel function $\mathbf{K}(x) = x^{\alpha}$. (a) is the image in the Cartesian coordinates, (b)–(e) are the corresponding images in the log-polar coordinates with varying kernel parameter α. Matching colors in the log-polar and Cartesian coordinates show the same regions.

then the inverse transformation can be formulated as Equation 2.5,

$$\begin{aligned} x'' &= \exp(Au)\cos(Bv) \\ y'' &= \exp(Au)\sin(Bv) \end{aligned} \tag{2.5}$$

To understand how the resolution changes in the log-polar space, consider $r = \|x, y\|_2 = \exp(Au)$.

Now, $\mathrm{d}r$ represents the change in r based on u,

$$\mathrm{d}r = A \cdot \exp(Au)\,\mathrm{d}u. \tag{2.6}$$

Inversely, $\mathscr{D}$ is defined as the number of pixels in the LP-buffer that map to a single pixel on the screen,

$$\mathscr{D} = \frac{\mathrm{d}u}{\mathrm{d}r} = \frac{1}{A} \cdot \exp(-Au). \tag{2.7}$$

Equation 2.7 shows the foveation effect of pixel density decreasing from the fovea to the periphery. In this formulation, it is not easy to systematically alter the density fall-off function and evaluate the perceptual quality of foveated rendering.

I propose a kernel log-polar mapping algorithm that allows us more flexibility to better mimic the fall-off of photo-receptor density of the human visual system,

$$\mathscr{D} = \frac{\exp\left(-wC\sigma \cdot \mathbf{K^{-1}}\left(\frac{u}{w}\right)\right)}{C\sigma \cdot \mathbf{K^{-1'}}\left(\frac{u}{w}\right)}. \tag{2.8}$$

Here, the constant parameter $C = \sqrt{1 + \left(\frac{H}{W}\right)^2}$ represents the ratio between screen diagonal and screen width. $\sigma = \frac{W}{w}$ represents the ratio between the full-resolution screen width and the reduced-resolution LP-buffer width, $\sigma^2 = \frac{W^2}{w^2}$ represents the ratio between the number of pixels in the full-resolution screen and the number of pixels in the reduced-resolution LP-buffer. Larger σ^2 corresponds to more

condensed LP-buffer, which means less calculation in the rendering process. A more condensed LP-buffer also means more foveation and greater peripheral blur.

The kernel function $\mathbf{K}(x)$ can be any monotonically increasing function with $\mathbf{K}(0) = 0$ and $\mathbf{K}(1) = 1$, such as the sum of power functions,

$$\mathbf{K}(x) = \sum_{i=0}^{\infty} \beta_i x^i, \quad \text{where} \sum_{i=0}^{\infty} \beta_i = 1. \tag{2.9}$$

Such kernel functions can be used to adjust the pixel density distribution in the LP-buffer. I use $\mathbf{K}(x) = \sum_{i=0}^{\infty} \beta_i x^i$ in this project because the calculation of power functions is fast on modern GPUs. There may be other kernel functions worth trying, such as $\mathbf{K}(x) = sin(x \cdot \frac{\pi}{2})$ and $\mathbf{K}(x) = \frac{e^x - 1}{e - 1}$. For example, for $C = \sqrt{2}$ and $\mathbf{K}(x) = x^\alpha$, the relationship between $\mathscr{D}$ and r under varying σ^2 and α is illustrated in Figure 2.13 [1]. Kernel functions can adjust the pixel density such that the percentage of the peripheral regions in the LP-buffer increases as shown in Figure 2.14 (c), (d), and (e). This makes it possible to increase the peripheral image quality while maintaining the same frame rates. A comparison among different kernel functions is shown in Figure 2.15 with $\sigma = 1.8$ and $C = \sqrt{2}$. The use of the kernel function reduces the artifacts in the zoomed-in peripheral view, improving the peripheral image quality.

Meanwhile, as shown in Figure 2.16, when $\alpha \geq 5.0$, the sampling rate of even the foveal region drops, affecting the visual quality of the fovea. A comparison among different σ is shown in Figure 2.17 with fixed $\alpha = 4.0$, $C = \sqrt{2}$.

[1] The figure is the visualization of sampling rate rather than the true sampling map.

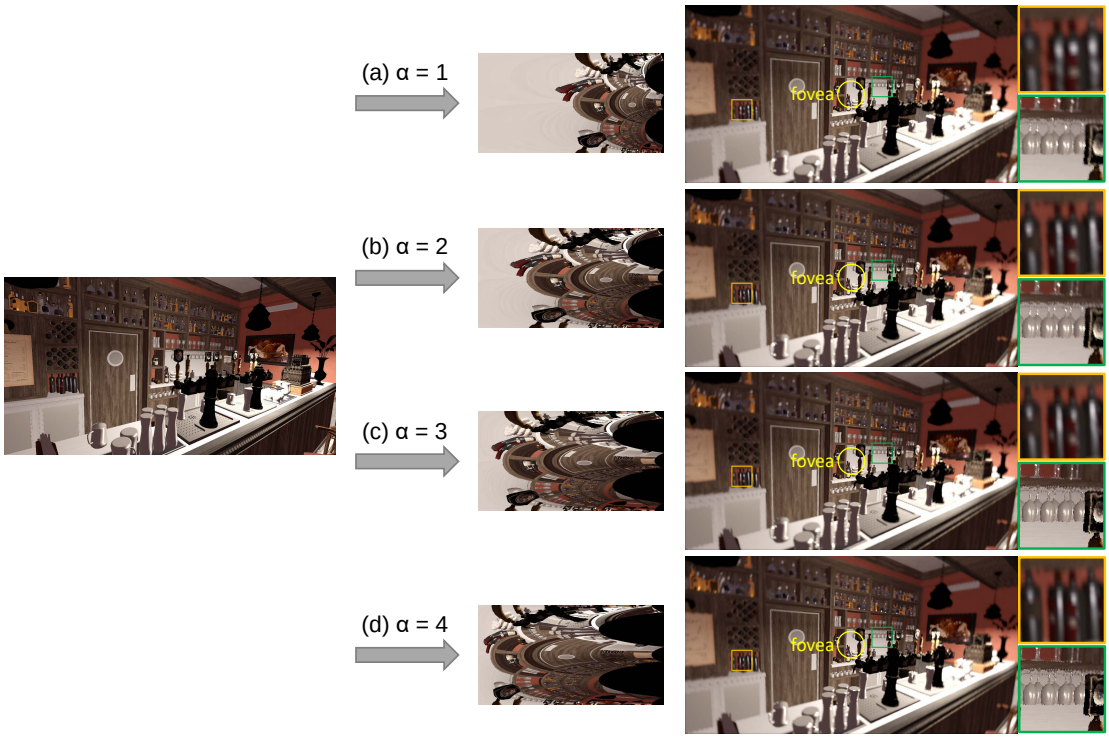


Figure 2.15: Comparison of foveated rendering with varying α for 2560×1440 resolution. From left to right: original rendering, kernel log-polar rendering, and the foveated rendering with zoomed-in view of the peripheral regions. Here $\sigma = 1.8$, (a) classic log-polar transformation, i.e. $\alpha = 1.0$, (b) kernel function with $\alpha = 2.0$, (c) kernel function with $\alpha = 3.0$, and (d) kernel function with $\alpha = 4.0$. The foveated rendering is at 67 FPS while the original is at 31 FPS.

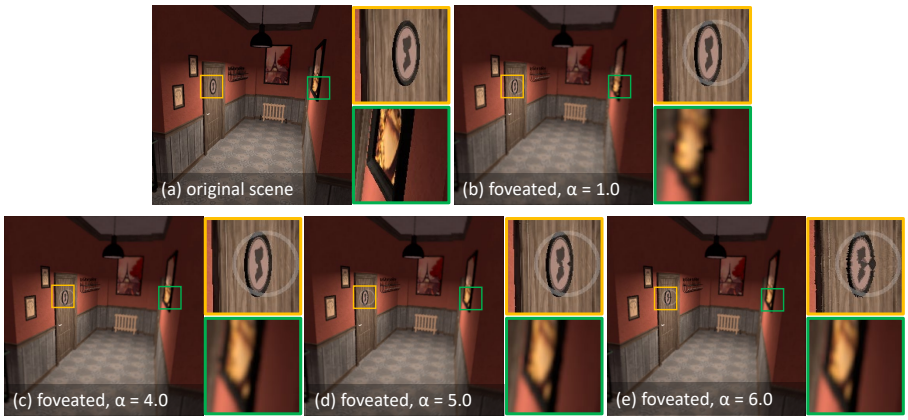


Figure 2.16: Comparison of foveated frame with different α (fovea is marked as the semi-transparent ring in the zoomed-in view): (a) original scene, (b) foveated with $\alpha = 1.0$, (c) foveated with $\alpha = 4.0$, (d) foveated with $\alpha = 5.0$, and (e) foveated with $\alpha = 6.0$. The lower zoomed-in views show that large α enhances the peripheral detail; the upper zoomed-in views show that when $\alpha \geq 5.0$, foveal quality suffers.

2.3.1 Pass I: Forward Kernel Log-polar Transformation

Kernel Log-polar Transformation For each pixel in screen space with coordinates (x, y), foveal point $\mathbf{F}(\mathring{x}, \mathring{y})$ in Cartesian coordinates, I change Equation 2.1 to Equation 2.10,

$$\begin{aligned} u &= \mathbf{K^{-1}} \left(\frac{\log \|x', y'\|_2}{L} \right) \cdot w \\ v &= \left(\arctan\left(\frac{y'}{x'} \right) + \mathbf{1}\left[y' < 0 \right] \cdot 2\pi \right) \cdot \frac{h}{2\pi} \end{aligned} \tag{2.10}$$

Here,

$$x' = x - \mathring{x}, \quad y' = y - \mathring{y}. \tag{2.11}$$

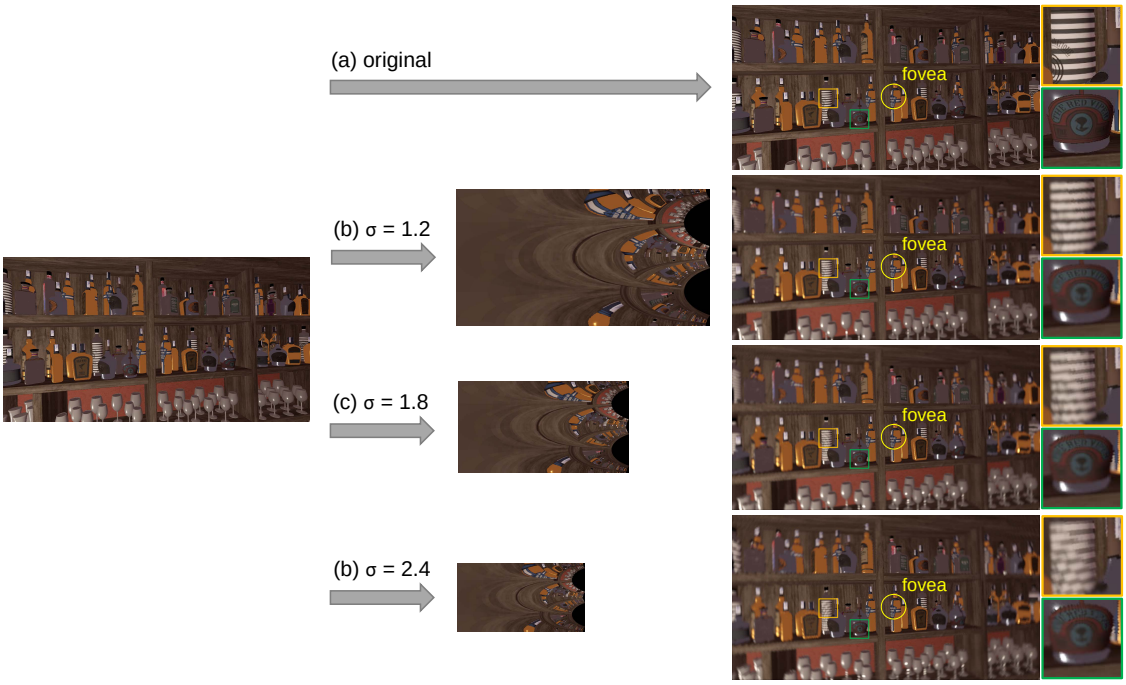


Figure 2.17: Comparison of foveated rendering with varying σ for 2560×1440 resolution. From left to right: original rendering, kernel log-polar rendering, the recovered scene in Cartesian coordinates, and a zoomed-in view of peripheral regions. Here, $\mathbf{K}(x) = x^4$, (a) full-resolution rendered at 31 FPS, (b) $\sigma = 1.2$ at 43 FPS, (c) $\sigma = 1.8$ at 67 FPS, and (d) $\sigma = 2.4$ at 83 FPS.

$\mathbf{K^{-1}}(\cdot)$ is the inverse of the kernel function, and L is the log of the maximum distance from fovea to one of the four corners of the screen as shown in Equation 2.12,

$$L = \log\left(\max\left(\max\left(l_1, l_2\right), \max\left(l_3, l_4\right)\right)\right). \tag{2.12}$$

Here,

$$\begin{aligned} l_1 &= \|\mathring{x}, \mathring{y}\|_2 \\ l_2 &= \|W - \mathring{x}, H - \mathring{y}\|_2 \\ l_3 &= \|\mathring{x}, H - \mathring{y}\|_2 \\ l_4 &= \|W - \mathring{x}, \mathring{y}\|_2 \end{aligned} \tag{2.13}$$

Lighting In lighting calculation for traditional deferred shading, mesh positions, normals, depth and material information such as roughness, index of reflection, and normal maps are fetched from the G-buffer [32, 33]. Instead of obtaining information from the G-buffer with texture coordinates (x, y), in my approach, I sample from the transformed kernel log-polar texture coordinates (u, v) with bilinear filtering for the lighting resources in the G-buffer. The reduced-resolution of the log-polar (LP) buffer helps in reducing the lighting calculation to only those pixels that matter in the final foveated rendering.

Internal Anti-aliasing Due to the low-resolution of the LP-buffer, there may be artifacts in the peripheral regions after the inverse transformation. However, I can directly perform denoising in the log-polar space. To reduce artifacts in the peripheral regions, I apply a Gaussian filter with a 3×3 kernel for the first quarter from the right of the rendering (corresponding to the peripheral regions) in the

Algorithm 1 Kernel Log-polar Transformation

Input:

Fovea coordinates in screen space: $(\mathring{x}, \mathring{y})$,

pixel coordinates in screen space: (x, y).

Output:

Pixel coordinates in the log-polar space: (u, v).

acquire fovea coordinates $(\mathring{x}, \mathring{y})$

for $x \in [0, W]$ **do**

  for $y \in [0, H]$ **do**

    $x' = x - \mathring{x}$

    $y' = y - \mathring{y}$

    $u = \mathbf{K^{-1}} \left(\frac{\log \|x', y'\|}{L} \right) \cdot w$

    $v = \left(\arctan \left(\frac{y'}{x'} \right) + \mathbf{1} \left[y' < 0 \right] \cdot 2\pi \right) \cdot \frac{h}{2\pi}$

  end for

end for

LP-buffer. Since the LP-buffer pixels correspond to the adaptive detail of foveated rendering, the Gaussian filtering in the LP-buffer gives us higher-level of anti-aliasing in the peripheral regions.

2.3.2 Pass II: Inverse Kernel Log-Polar Transformation

Pass II performs the inverse kernel log-polar transformation to Cartesian coordinates, applies anti-aliasing, and renders to screen.

Inverse Kernel Log-polar Mapping Transformation I can recover the Cartesian coordinates (x'', y''), from the pixel coordinates (u, v) and the fovea coordinates $(\mathring{x}, \mathring{y})$ using Algorithm 2 with bilinear filtering.

Post Anti-aliasing One of the crucial considerations in foveated rendering is mitigating temporal artifacts due to aliasing in the peripheral, high eccentricity regions. I apply temporal anti-aliasing (TAA) [56] with Halton sampling [59] to the recovered screen-space pixels after the inverse kernel log-polar transformation. I also use Gaussian filtering with different kernel sizes η for different L (as defined in Equation 2.12) in post anti-aliasing. The kernel size η is shown in Equation 2.14, which depends on the normalized distance between the pixel coordinate and the fovea,

$$\eta = 3 + 2 \times \left\lfloor \frac{\frac{\|x', y'\|_2}{e^L} - 0.10}{0.05} \right\rfloor. \tag{2.14}$$

Algorithm 2 Kernel Log-polar Inverse Transformation

Input:

Fovea coordinates in screen space: $(\mathring{x}, \mathring{y})$,

pixel coordinates in the log-polar coordinates: (u, v).

Output:

Screen-space coordinates (x'', y'') for pixel coordinates (u, v).

update L with fovea coordinates $(\mathring{x}, \mathring{y})$ with Equation 2.12

let $A = \frac{L}{w}$, $B = \frac{2\pi}{h}$

for $u \in [0, w]$ **do**

  for $v \in [0, h]$ **do**

    $x'' = \exp(A \cdot \mathbf{K}(u)) \cdot \cos(Bv) + \mathring{x}$

    $y'' = \exp(A \cdot \mathbf{K}(u)) \cdot \sin(Bv) + \mathring{y}$

  end for

end for

2.4 User Study

I have carried out user studies to empirically establish the most suitable foveation parameter values for σ and α that result in visually acceptable foveated rendering. To systematically investigate this, I conducted a pilot study to examine a broad range of the two parameters, σ^2 and α. I used the results and my experience with the pilot study to fine tune the protocol and ranges of σ^2 and α for the final user study.

2.4.1 Apparatus

My user study apparatus, shown in Figure 2.18, consists of an *Alienware* laptop with an *NVIDIA GeForce GTX 1080*, a *FOVE* head-mounted display, and an *XBOX* controller. The *FOVE* display has a 100° field of view, a resolution of 2560 × 1440, and a 120 Hz infrared eye-tracking system with a precision of 1° and a latency of 14

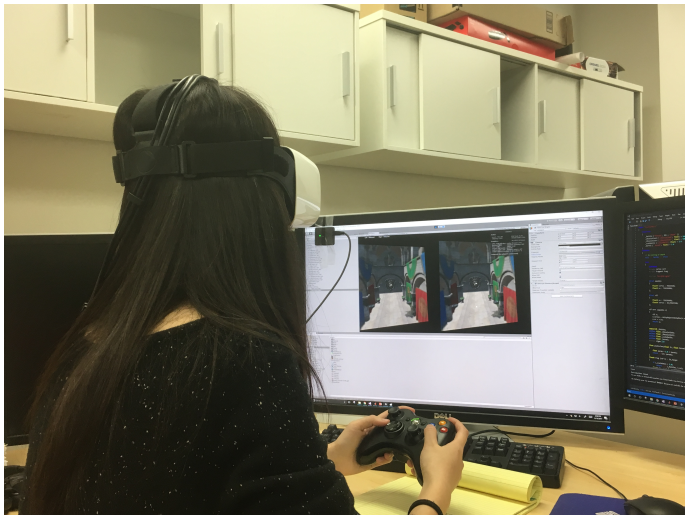
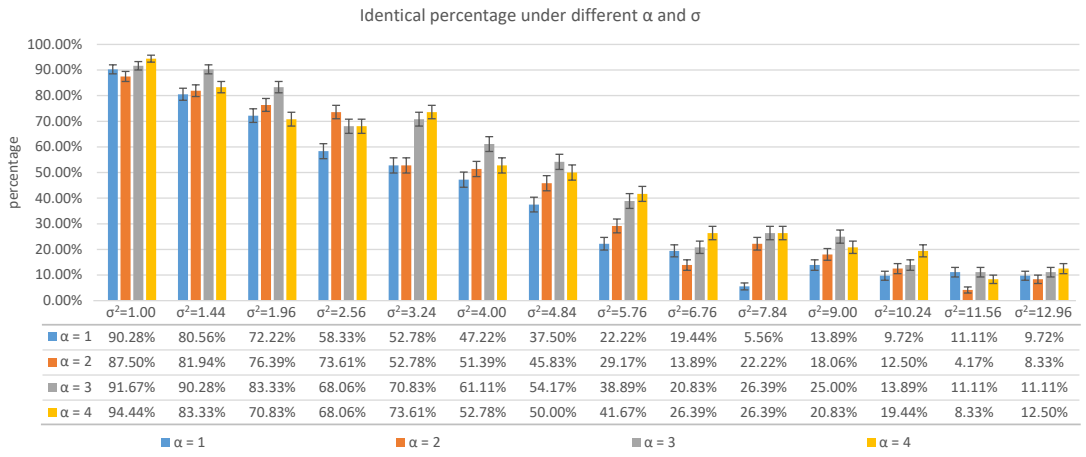


Figure 2.18: User study setup.



	σ²=1.00	σ²=1.44	σ²=1.96	σ²=2.56	σ²=3.24	σ²=4.00	σ²=4.84	σ²=5.76	σ²=6.76	σ²=7.84	σ²=9.00	σ²=10.24	σ²=11.56	σ²=12.96
α = 1	90.28%	80.56%	72.22%	58.33%	52.78%	47.22%	37.50%	22.22%	19.44%	5.56%	13.89%	9.72%	11.11%	9.72%
α = 2	87.50%	81.94%	76.39%	73.61%	52.78%	51.39%	45.83%	29.17%	13.89%	22.22%	18.06%	12.50%	4.17%	8.33%
α = 3	91.67%	90.28%	83.33%	68.06%	70.83%	61.11%	54.17%	38.89%	20.83%	26.39%	25.00%	13.89%	11.11%	11.11%
α = 4	94.44%	83.33%	70.83%	68.06%	73.61%	52.78%	50.00%	41.67%	26.39%	26.39%	20.83%	19.44%	8.33%	12.50%

Figure 2.19: The percentage of times that the participants considered the foveated rendering and the full-resolution rendering to be the same for varying σ^2 and α in pilot user study with 24 participants.

ms, the system latency meets the eye tracking delay requirement of 50 ms - 70 ms.

2.4.2 Pilot Study

Procedure The session for each participant lasted between $35 - 50$ minutes and involved four stages: introduction, calibration, training, and testing.

In the introduction stage, I showed participants the *FOVE* headset, the eye trackers, and the *XBOX* controllers and discussed how to use them. I did not provide any information about the research or the algorithm to avoid biasing the participants towards any rendering.

After the participant comfortably wore the HMD, I moved forward to the calibration stage, where I ran a one-minute eye-tracking calibration program provided by the *FOVE* software development kit.

In the training stage, I presented the participants with 20 trials with different

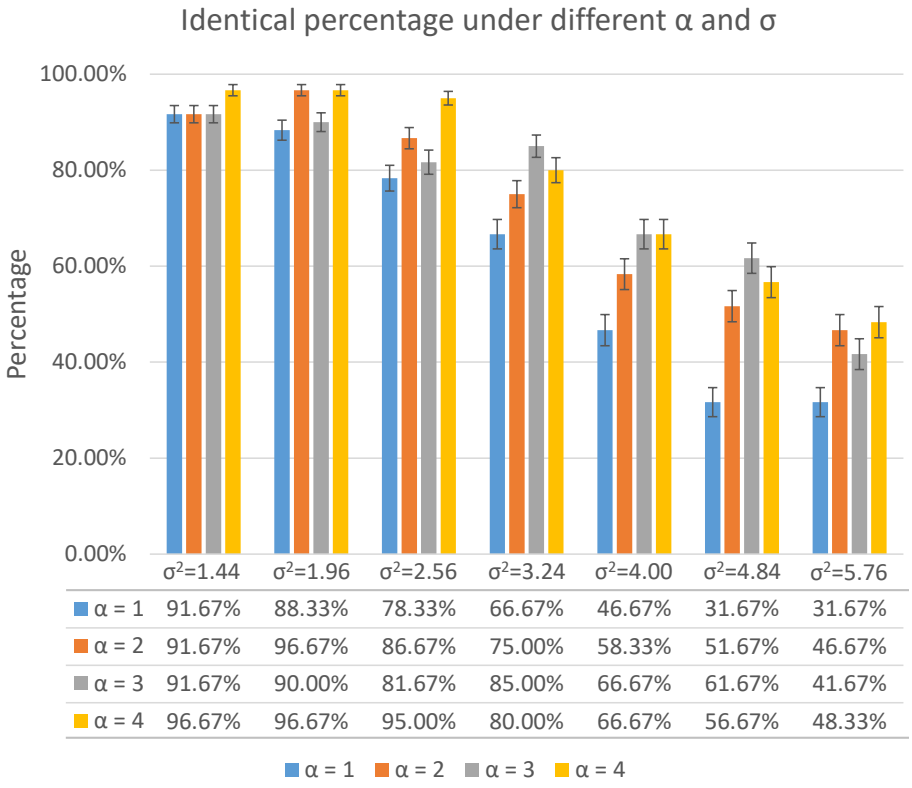


Figure 2.20: The percentage of times that participants considered the foveated rendering and the full-resolution rendering to be identical for different σ^2 and α in the final user study with 18 participants.

combinations of σ^2 and α, to ensure that they are familiar with the HMD and the controller.

Trials in the training and testing stages were identical. In each trial of the two-alternative forced choice test, I presented participants with a pair of rendered scenes, each for 2 seconds and separated by a black-screen interval of 0.75 seconds. One scene uses full-resolution rendering, and the other uses KFR with different parameters σ^2 and α. In each trial, I presented the KFR scene and the full-resolution scene in a random order. The participant indicated whether the two images look the

same by pressing a button on the *XBOX* controller. I instructed the participants to maintain their gaze at the center of the screen, even though the foveated renderer can use eye-tracking to update the foveated image.

The testing stage had three sessions, each with 56 trials. The LP-buffer resolution reduction parameter σ ranges from 1.0 to 3.6 with step size 0.2 (σ^2 ranges from 1.00 to 12.96), and the kernel sampling distribution parameter α ranges from 1 to 4 with step size 1. I rendered scenes from the *Sponza* and *Amazon Lumberyard Bistro* datasets for different sessions. I allowed the participants to have some rest between different sessions.

Participants In the pilot study, I recruited 24 participants via campus email lists and flyers. All participants are at least 18 years old with normal or corrected-to-normal vision (with contact lenses). The participants are collected using campus flyers and emails.

Results and Analysis I define P_I as the percentage of the trials for which participants reported the two images shown in a trial to be the same. The results of P_I are shown in Figure 2.19. First, I find that P_I is inversely related to σ^2. With increase in σ^2, the LP-buffer gets smaller, thus reducing the overall sampling rate in foveated rendering. Second, I notice that with the increase of α, P_I significantly increases for σ ranging from 1.2 to 2.8 (σ^2 ranging from 1.44 to 7.84). This shows that for the same σ, the perception of the quality of foveated rendering increases by the use of α for kernel functions. For $\sigma = 1.0$ and $\sigma > 2.8$ the improvement is not

significant. The reason is that for $\sigma = 1.0$, the foveated renderings for $\alpha = 1$ and $\alpha > 1$ are both clear, there is little space for improvement. Similarly, for $\sigma > 2.8$, even if the quality improves by applying kernel functions, it still looks blurry for both images. Therefore, the participants choose "different" for these comparisons. Third, some participants reported that the length of the study led to visual fatigue and that they were not sure about some of their responses.

Using the above observations, I modified the final user study to be shorter and more focused.

First, to reduce the total time that participants are in the HMD, I used the fact that most participants found foveated renderings different from the full-resolution rendering for $\sigma > 2.4$ ($\sigma^2 > 5.76$). Since the goal is to accelerate rendering while maintaining perceptually similar quality, I reduced the range of σ to be between 1.2 to 2.4 (σ^2 between 1.44 to 5.76) in the final user study.

Second, I observed that the participants quickly came up to speed within a couple of trials in the training session. I therefore reduced the number of trials in the training session from 20 to 5. This also allowed us to shorten the user study duration and maintain a high level of visual attentiveness of the participants. Third, some of the participants reported that the rendering time of 2 seconds was too short. To address this I increased the time of each rendering to 2.5 seconds in the final study. Fourth, to continually check for the visual attentiveness of the participants, I modified the final user study by randomly inserting 30% of the trials to be "validation trials" that had identical full-resolution renderings for both choices. If the participant declared these validation renderings to be different, I would ask the participant to

stop, get some rest, and then continue. After making these changes, the total time participants spent in the HMD was reduced from around 25 minutes in the pilot study to around 15 minutes in the final study.

2.4.3 Final User Study

Procedure The introduction and calibration stages are the same as the pilot user study. The training session includes five trials with different parameters. Each testing session involves 28 trials with multiple parameter combinations (parameter σ ranging from 1.2 to 2.4 with the step size 0.2 (σ^2 ranging from 1.44 to 5.76); and kernel parameter α ranging from 1 to 4 with the step size 1) as well as additional "validation trials". Order of the parameters is fully counterbalanced. The participants are asked to rest after each session or if they do not pass a "validation trial". I also changed the rendering-display time to 2.5 seconds.

Participants I recruited 18 participants via campus email lists and flyers. All participants were at least 18 years old with normal or corrected-to-normal vision (with contact lenses). The participants are collected using campus flyers and emails.

Results and Analysis I report the percentage P_I and the corresponding standard error in Figure 2.20. I make the null hypothesis (H_0) that the foveated rendering results with the four kernel functions are equally effective. As shown in Table 1, with a Cochran's Q test [60,61], I have found that there exists a significant difference across the multiple α for $\sigma = 1.6, 1.8, 2.2$ ($\sigma^2 = 2.56, 3.24, 4.84$) with $\chi^2(3) = 7.81, p < 0.05$.

The results with very small $\sigma = 1.2, 1.4$ ($\sigma^2 = 1.44, 1.96$) and very large $\sigma = 2.4$ ($\sigma^2 = 5.76$) are not significantly different, which are reasonable. For small σ^2, the rendering result without kernel function is clear enough, so there is little room for improvement. For large σ^2, both the rendering results with and without kernel function are blurry for the users.

Table 2.1: Cochran's Q values at different σ^2.

σ^2	1.44	1.96	2.56	3.24	4.00	4.84	5.76
Cochran's Q value	1.72	5.79	8.20	8.25	7.49	14.27	5.48
p value	0.631	0.122	0.042	0.041	0.058	0.002	0.139

To achieve visually acceptable results for foveated rendering, I use a threshold of 80% responses considering foveated rendering to be visually indistinguishable from full-resolution rendering. To achieve the highest rendering acceleration, I look for the highest σ that met this threshold. I therefore choose $\sigma = 1.8$ ($\sigma^2 = 3.24$) and $\alpha = 4$ as the desired parameters for the interactive rendering evaluation.

2.5 Results and Acceleration

I implemented kernel foveated rendering on *NVIDIA GeForce GTX 1080*, by using the deferred shading pipeline of the *Falcor* engine [62]. I report results of my rendering acceleration for resolutions of 1920×1080, 2560×1440, and 3840×2160. Using the results from the final user study, I selected the LP-buffer parameter $\sigma = 1.8$, and kernel parameter $\alpha = 4$ for the evaluations below.

2.5.1 Rendering Acceleration of 3D Textured Meshes

I use the *Amazon Lumberyard Bistro* [19] scene with physically-based shading, reflection, refraction, and shadows to simulate the complex shading effects as shown in Figure 2.21. I choose *Amazon Lumberyard Bistro* because this scene has complex triangular meshes, rendering textures and compact lighting effect. The comparison of the break-down of rendering time between KFR and the ground truth of deferred shading is shown in Table 2. I observed that the frame rate increases for all resolutions as shown in Table 3, with a speedup of $2.0X - 2.8X$.

2.5.2 Rendering Acceleration of Ray-casting Rendering

Ray casting uses ray–surface intersection tests to solve a variety of problems in computer graphics and computational geometry. It can also be used for creating scenes. Rendering of high-resolution ray cast scenes can be an extremely time-consuming process. I used the complex ray-casting scene with 16 different primitives by Íñigo Quílez to evaluate the acceleration of kernel foveated rendering. Figure 2.22 shows a comparison of the foveated scene and the ground truth. The frame rate increases for all resolutions as shown in Table 3, with a speedup of $2.9X - 3.2X$.

Figure 2.21: Comparison of (a) full-resolution rendering and (b) foveated rendering for 3D meshes involving a geometry pass with $1,020,895$ triangles as well as multiple G-buffers

at 2560×1440 resolution.

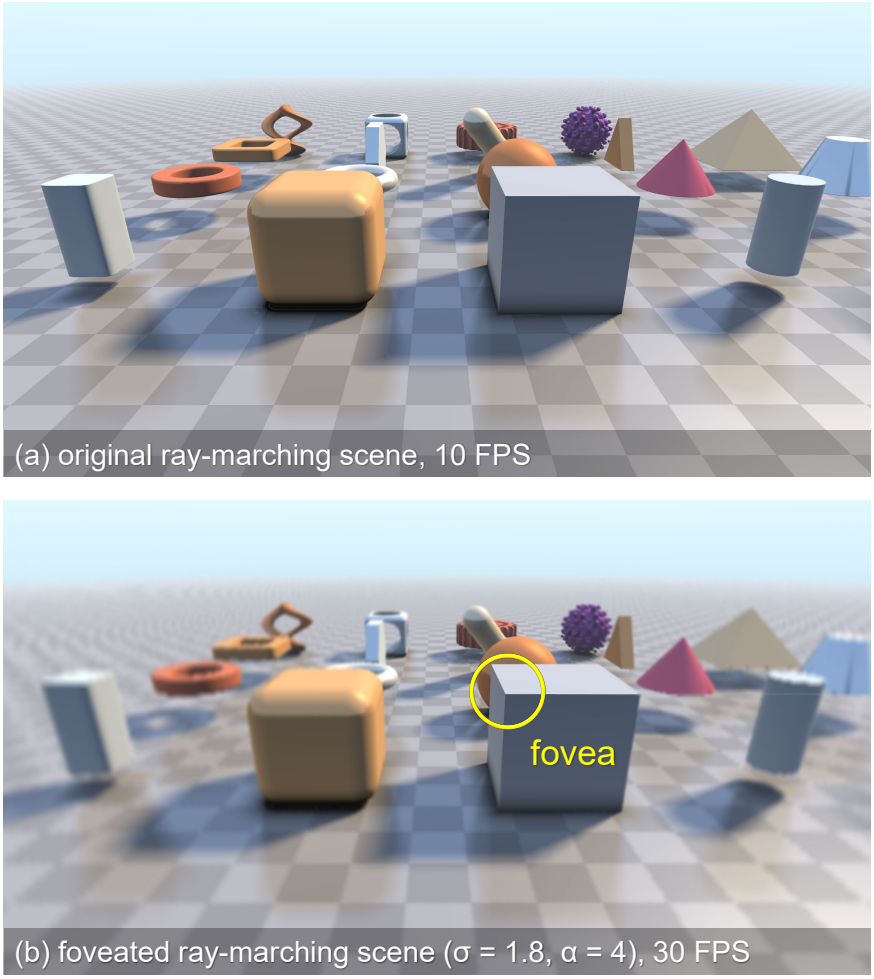


Figure 2.22: Comparison of (a) full-resolution rendering and (b) foveated ray-marching scene with 16 samples per pixel

rendered at 2560×1440.

Table 2.2: Timing comparison between the ground truth and KFR for one frame. The resolution is 1920×1080.

Procedure	Timing (ms)	
	Ground Truth	KFR
Depth Pass	0.327	0.309
Shadow Pass	3.744	4.503
Defer Pass	2.985	3.034
SkyBox	0.039	0.039
Shading / Pass1	22.043	6.674
Pass2	N/A	0.090
Total	29.138	14.649
Total GPU Time	31.892	17.052

2.6 Discussion

Here I compare the Kernel Foveated Rendering (KFR) pipeline with selected prior art, including: *Foveated 3D Graphics* (F3D) [6], *Multi-rate Shading* (MRS) [11], *Coarse Pixel Shading* (CPS) [7], and *Adaptive Image-Space sampling* (AIS) [13].

As mentioned by [13], both MRS and CPS pipelines require adaptive shading features which are not yet commonly available on commodity GPUs and so they

Table 2.3: Frame rate and speedup comparison for kernel foveated rendering at different resolutions with $\sigma = 1.8$, $\alpha = 4.0$.

Scene	3D Textured Meshes			Ray Casting		
Resolution	Ground Truth	Foveated	Speedup	Ground Truth	Foveated	Speedup
1920×1080	55 FPS	110 FPS	2.0X	20 FPS	57 FPS	2.9X
2560×1440	31 FPS	67 FPS	2.2X	10 FPS	30 FPS	3.0X
3840×2160	8 FPS	23 FPS	2.8X	5 FPS	16 FPS	3.2X

rely on software simulator implementations. In contrast, F3D, AIS, and the KFR pipelines can be easily mapped onto the current generation of GPUs.

The F3D pipeline has achieved impressive speedups of $10X-15X$ in the informal user study, and a factor of $4.8X - 5.7X$ in the formal user study. Nevertheless, the F3D approach uses three discrete layers, while the KFR parameterizes the distribution of samples continuously in the log-polar domain. F3D relies on specifically designed anti-aliasing algorithms including jitter sampling and temporal reprojection, thus limiting F3D to simpler material models and less complex geometry [13]. In contrast, KFR could easily be coupled with the state-of-the-art screen-space anti-aliasing techniques, such as TAA [56] and recent G-buffer anti-aliasing strategies [57].

Both the AIS and KFR pipelines mimic the continuously changing distribution of photo-receptors in the retina. Nonetheless, there are three significant differences: complexity and evaluation of the perceptual model, interpolation, and speedup. First, AIS uses four parameters from [63] to approximately model the linear degradation

behavior of acuity with 30° eccentricity. However, these parameters have not yet been evaluated on how they affect foveation and perception in HMDs or beyond 30° eccentricity. In contrast, KFR uses only two parameters: the reduced-resolution LP-buffer parameter σ and the kernel parameter α in conjunction with a simple coordinate transformation. KFR has established desirable values of α and σ through user studies in head-mounted displays. Second, AIS relies on the pull-push interpolation [54] to fill the pixels that are missed due to variable sampling of silhouette features and object saliency.

In comparison, KFR uses the built-in GPU-driven mipmap interpolation which reduces the additional interpolation cost. However, it is worth investigating how incorporating object saliency [64, 65] could further improve the KFR pipeline.

It is a challenge to compare multiple foveated approaches given the varying hardware and perceptual quality of the results. One possibility is to compare the speedups as percentages of rendering time reduction with certain reduction sampling rate. By rendering with 59% of the total amount of shaded pixels, AIS reports an overall rendering time reduction of 25.4%, while KFR achieves 29.9% reduction on average. Like AIS, KFR speedup also depends on the amount of time spent in shading; the greater the shader computations, the higher will be the KFR speedup.

Chapter 3: 3D-Kernel Foveated Rendering for Light Fields

3.1 Overview

Classic light field rendering is limited to low-resolution images because the rendering process of large-scale, high-resolution light field image arrays requires a great amount of texture sampling, thus increasing the latency of rendering and streaming.

With the rapid advances in optical microscopy imaging, several technologies have been developed for interactively visualizing [66] or reconstructing microscopy volumes [67–71]. Recently, light field microscopy [72] has emerged, which allows one to capture light fields of biological specimens in a single shot. Afterwards, one could interactively explore the microscopic specimens with a light-field-renderer, which automatically generates novel perspectives and focal stacks from the microscopy data [73]. Unlike regular images, light field microscopy enables natural-to-senses stereoscopic visualization. Users may examine the high-resolution microscopy light fields with the inexpensive commodity virtual reality head-mounted displays (HMDs) as a natural stereoscopic tool. The method of rendering light field microscopy can be applicable to high-resolution light fields from other sources.

To the best of our knowledge, the interactive visualization of high-resolution

light fields with low latency and high-quality remains a challenging problem.

Human vision spans 135° vertically and 160° horizontally, but the highest-resolution foveal vision only covers a 5° central region of the vision [6]. As estimated by Patney *et al.* [8], only 4% of the pixels in a modern HMD are mapped on the fovea. Therefore, foveated rendering techniques that allocate more computational resources for the foveal pixels and fewer resources elsewhere can dramatically speed up light field visualization.

In this chapter, we present 3D-kernel foveated rendering (3D-KFR), a novel approach to extend the kernel foveated rendering (KFR) [23] framework to light fields. In 3D-KFR, we parameterize the foveation of light fields by embedding polynomial kernel functions in the classic log-polar mapping [28, 29] for each slice. This allows us to alter both the sampling density and distribution, and match them to human perception in virtual reality HMDs. Next, we optimize 3D-KFR by adjusting the weight of each slice in the light fields, so that it automatically selects the optimal foveation parameters for different images according to the gaze position and achieves higher speedup. In this way, 3D-KFR further accelerates the rendering process of high-resolution light fields while preserving the perceptually accurate foveal detail.

We have validated our approach on the rendering of light fields by carrying out both quantitative experiments and user studies. Our method achieves speedups of $3.47\times - 7.28\times$ on different levels of foveation and different rendering resolutions. Moreover, our user studies suggest the optimal parameters that anyone can use for rendering of foveated light fields on modern HMDs.

In summary, our contributions include:

1. designing 3D-KFR, a new visualization method to observe the light fields, which provides similar visual results as the original light field, but at a higher rendering frame rate;

2. conducting user studies to identify the 3D-KFR parameters governing the density of sampling to maximize perceptual realism and minimize computation for foveated light fields in HMDs;

3. implementing the 3D-KFR light field pipeline on a GPU, and achieving speedups of up to 7.28× for the light field with a resolution of $25 \times 25 \times 1024 \times 1024$ px with minimal perceptual loss of detail.

3.2 Related Work

3.2.1 Light Field Rendering

4D light fields [16, 54] represent an object or a scene from multiple camera positions as shown in Figure 3.1. We can generating new views from arbitrary camera positions without depth information or feature matching, simply by interpreting the input images as 2D slices of a 4D light field function, which completely characterizes the flow of light through unobstructed space in a static scene with fixed illumination.

Chai *et al.* [74] determine the minimum sampling rate for light field rendering by spectral analysis of light field signals using the sampling theorem. Ng [75] contributes to a Fourier-domain algorithm for fast digital focusing for light fields. Lanman and Luebke [76] propose near-eye light field displays supporting continuous

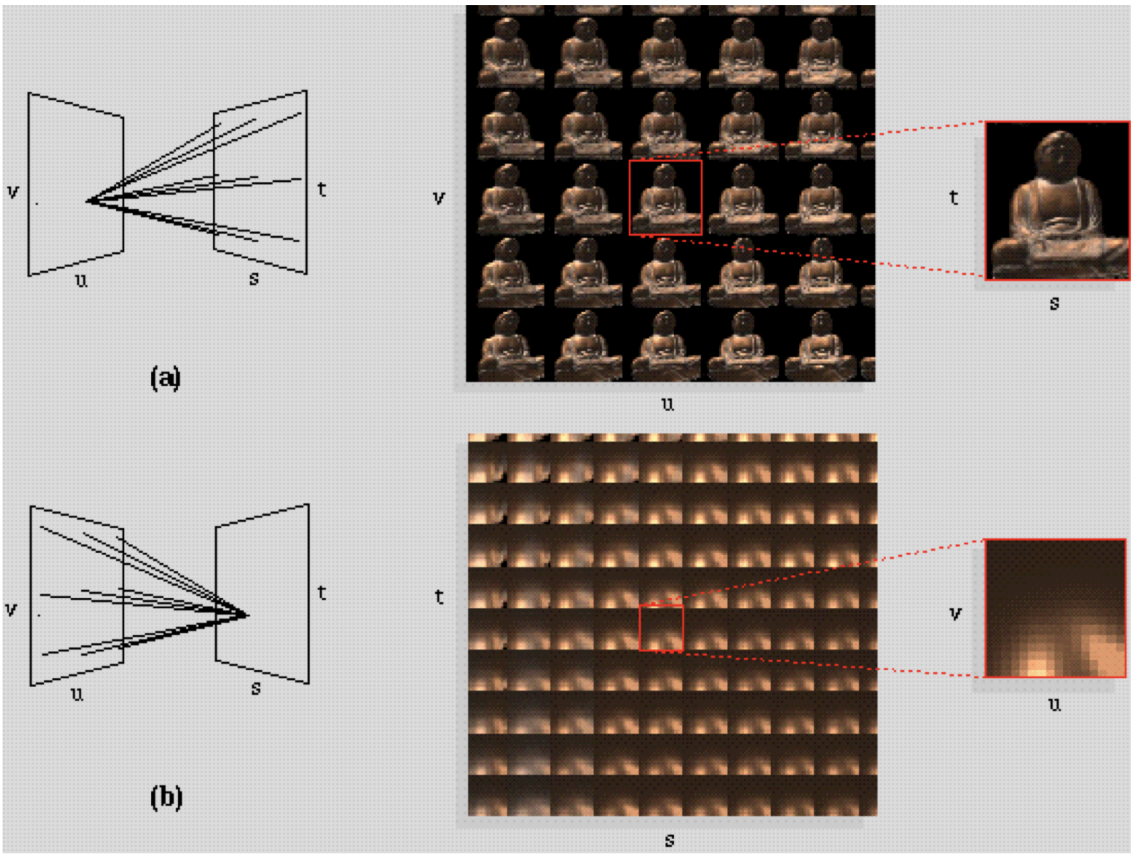


Figure 3.1: Levoy and Hanrahan [16] two visualizations of a light field. (a) Each image in the array represents the rays arriving at one point on the *uv* plane from all points on the *st* plane, as shown at left. (b) Each image represents the rays leaving one point on the *st* plane bound for all points on the *uv* plane. The images in (a) are off-axis perspective views of the scene, while the images in (b) look like reflectance maps.

accommodation of the eye throughout a finite depth of field, thus providing a means to address the accommodation-convergence conflict occurring with existing stereoscopic displays. Huang *et al.* [77] analyze the lens-distortion in light field rendering and correct it, thus improving the resolution and blur quality. Zhang *et al.* [78] propose a unified mathematical model for multilayer-multiframe compressive light field displays that significantly reduces artifacts compared with attenuation-based multilayer-multiframe displays. Lee *et al.* [79] propose foveated retinal optimization (FRO), which has tolerance for pupil movement without gaze tracking while maintaining image definition and accurate focus cues. The system achieves $38^\circ \times 19^\circ$ FoV, continuous focus cues, low aberration, small form factor, and clear see-through property. However, FRO adopts the idea of foveation to improve the display performance of the multi-layer displays rather than the rendering speed of 3D content. Sun *et al.* [17] design a real-time foveated 4D light field rendering and display system. Their work analyzes the bandwidth bound for perceiving 4D light fields and proposes a rendering method with importance sampling and a sparse reconstruction scheme. Their prototype renders only $16\% - 30\%$ of the rays without compromising the perceptual quality. The algorithm is designed for the desktop screen. In contrast to the previous work, our approach focused on foveated visualization of large light fields in virtual reality HMDs.

Sun *et al.* [17] design a real-time foveated 4D light field rendering and display system as shown in Figure 3.2. Based on the theoretical analysis on visual and display bandwidths, they find the frequency bound for the retina, eye lens and the display. Afterwards, they finish a series of psychophysical experiments and formulate

a content-adaptive importance model in the 4D ray space. Their prototype renders only $16\% - 30\%$ of the rays without compromising the perceptual quality.

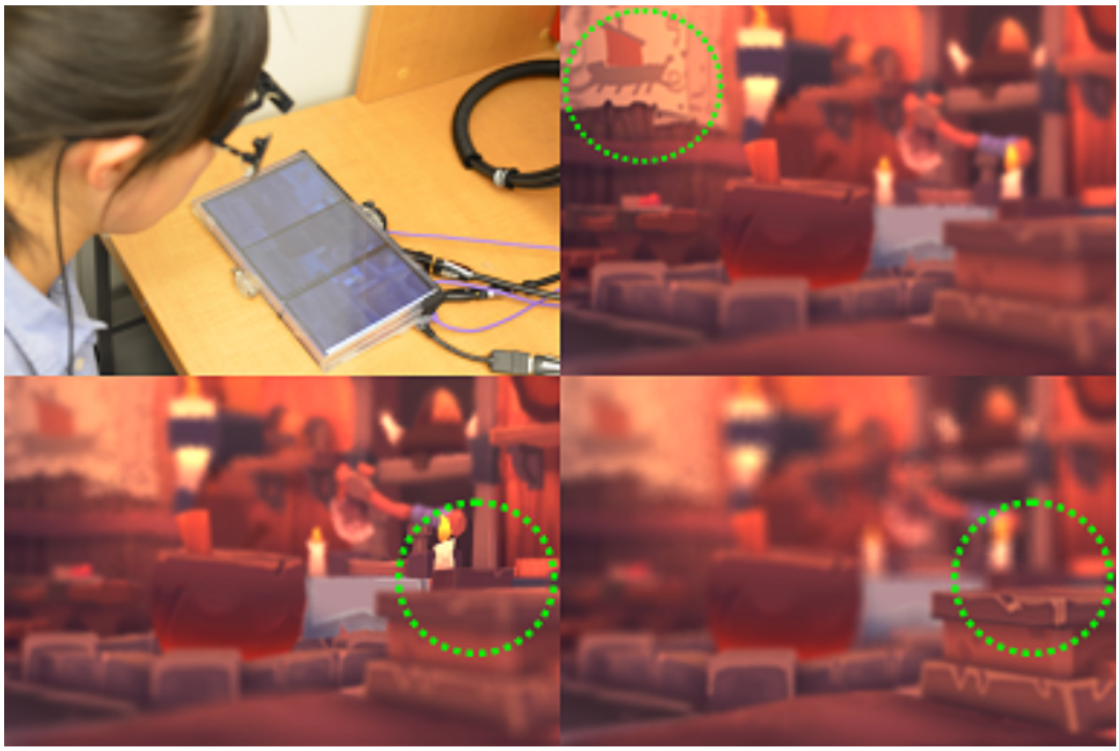


Figure 3.2: Sun *et al.* [17] design a real-time foveated 4D light field rendering and display system.

Mildenhall *et al.* [80] propose an algorithm to render novel views from an irregular grid of sampled views by expanding each sampled view into a local light field via a multiplane image (MPI) scene representation and blending adjacent local light fields.

3.2.2 Light Field Microscopy

Weinstein and Descour [81] use lens arrays for single-view-point array microscope with ultra-wide FOV instead of light fields with perspective views. Levoy *et al.* [72] propose using light fields to produce microscopy with perspective views and focal stacks. Wilt *et al.* [82] confirm the importance of observing cellular properties by using light microscopy for neuroscientists. The advances include enabling new experimental capabilities and permitting functional imaging at faster speeds. Prevedel *et al.* [83] implement a light field deconvolution microscopy and demonstrate its ability to simultaneously capture the neuronal activity of the entire nervous system.

3.3 Proposed Algorithm

In this section, we first introduce KFR for 4D light field rendering. Next, we generalize KFR to 3D-KFR. Finally, we discuss the resulting rendering acceleration that 3D-KFR is able to achieve over KFR.

3.3.1 KFR for 4D Light Field Rendering

In the $k \times k$ light field with image resolution of $W \times H$, the total number of texture samples for rendering the original light field N_{original} is,

$$N_{\text{original}} = k^2 \cdot WH \tag{3.1}$$

KFR accelerates the rendering process of light fields by reducing texture sampling. In the first pass, we perform kernel log-polar transformation for each

slice and render to a reduced resolution buffer with dimensions of $k \times k \times w \times h$. In the second pass, we perform the inverse log-polar transformation to map the pixels back to the screen. The kernel function $\mathbf{K}(x)$ is defined in [23], which can be any monotonically increasing function with $\mathbf{K}(0) = 0$ and $\mathbf{K}(1) = 1$, such as a polynomial,

$$\mathbf{K}(x) = \sum_{i=0}^{\infty} \beta_i x^i, \quad \text{where} \sum_{i=0}^{\infty} \beta_i = 1. \tag{3.2}$$

We next present the two passes of the KFR algorithm.

In the first pass, we transform the image from Cartesian coordinates to kernel log-polar coordinates. For each pixel in screen space with coordinates (x, y), foveal point $\mathbf{F}(\mathring{x}, \mathring{y})$ in Cartesian coordinates, we define x', y' as

$$x' = x - \mathring{x}, \quad y' = y - \mathring{y}. \tag{3.3}$$

Then, we transform point (x', y') to (u, v) in kernel log-polar coordinates using Equation 3.4,

$$\begin{aligned} u &= \mathbf{K^{-1}}\left(\frac{\log\|x', y'\|_2}{L}\right) \cdot w \\ v &= \left(\arctan\left(\frac{y'}{x'}\right) + \mathbf{1}\left[y' < 0\right] \cdot 2\pi\right) \cdot \frac{h}{2\pi} \end{aligned} \tag{3.4}$$

$\mathbf{K^{-1}}(\cdot)$ is the inverse of the kernel function, and L is the log of the maximum distance from fovea to one of the four corners of the screen as shown in Equation 3.5,

$$L = \log\left(\max\left(\max\left(l_1, l_2\right), \max\left(l_3, l_4\right)\right)\right). \tag{3.5}$$

Here,

$$\begin{aligned} l_1 &= \|\mathring{x}, \mathring{y}\|_2 \\ l_2 &= \|W - \mathring{x}, H - \mathring{y}\|_2 \\ l_3 &= \|\mathring{x}, H - \mathring{y}\|_2 \\ l_4 &= \|W - \mathring{x}, \mathring{y}\|_2 \end{aligned} \tag{3.6}$$

We define $\sigma = \frac{W}{w} = \frac{H}{h}$ as the ratio between the full-resolution screen width (or height) and the reduced-resolution buffer width (or height). The number of texture samples for the first pass $N_{\text{KFR pass 1}}$ can be theoretically inferred as:

$$N_{\text{KFR pass 1}} = k^2 \cdot \frac{W}{\sigma} \cdot \frac{H}{\sigma} = \frac{k^2}{\sigma^2} \cdot WH \tag{3.7}$$

In the second pass, a pixel with kernel log-polar coordinates (u, v) is transformed back to (x'', y'') in Cartesian coordinates. Let

$$A = \frac{L}{w}, \quad B = \frac{2\pi}{h}, \tag{3.8}$$

then the inverse transformation can be formulated as Equation 3.9,

$$\begin{aligned} x'' &= \exp\left(A \cdot \mathbf{K}\left(u\right)\right) \cdot \cos\left(Bv\right) + \mathring{x} \\ y'' &= \exp\left(A \cdot \mathbf{K}\left(u\right)\right) \cdot \sin\left(Bv\right) + \mathring{y} \end{aligned} \tag{3.9}$$

The number of texture samples for the second pass $N_{\text{KFR pass 2}}$ is

$$N_{\text{KFR pass 2}} = WH \tag{3.10}$$

The total number of texture samples for rendering the light field with KFR is

$$\begin{aligned} N_{\text{KFR}} &= N_{\text{KFR pass 1}} + N_{\text{KFR pass 2}} \\ &= (\frac{k^2}{\sigma^2} + 1) \cdot WH \end{aligned} \tag{3.11}$$

The parameter σ controls the total number of pixels of the reduced-resolution buffer, thus controlling the foveation rate and the amount of sampling. Comparing Equations 3.1 and 3.11, we notice that the number of texture samples can be greatly reduced in KFR with $\sigma > 1.0$. Kernel function controls the distribution of pixels in the whole image. By adjusting kernel functions, we can determine the pixel distribution and choose one that mimics the photo-receptor distribution of human eyes. The kernel log-polar mapping algorithm allows us to mimic the fall-off of photo-receptor density of human visual system with different σ and different kernel functions.

3.3.2 3D-KFR for 4D Light Field

The rendering of 4D light field is different from the rendering of 3D meshes because the center camera position of each slice is different. Since the foveation level of a pixel is affected by the distance to the center camera, the foveation parameter can be different for different slices in a light-field image array. We optimize the KFR algorithm into 3D-KFR by adjusting the weight of each slice in the light field, so that it is able to automatically select the optimal foveation parameters for different images according to the gaze position, thereby achieving greater speedup. Our algorithm consists of two passes as shown in Figure 1.4.

We define d as the distance between the local center camera of the frame $X_{\text{cam } ij}(x_{ij}, y_{ij})$ and the gaze position $X_{\text{cam } 0}(x_0, y_0)$,

$$d = \|X_{\text{cam } 0} - X_{\text{cam } ij}\|_2 \tag{3.12}$$

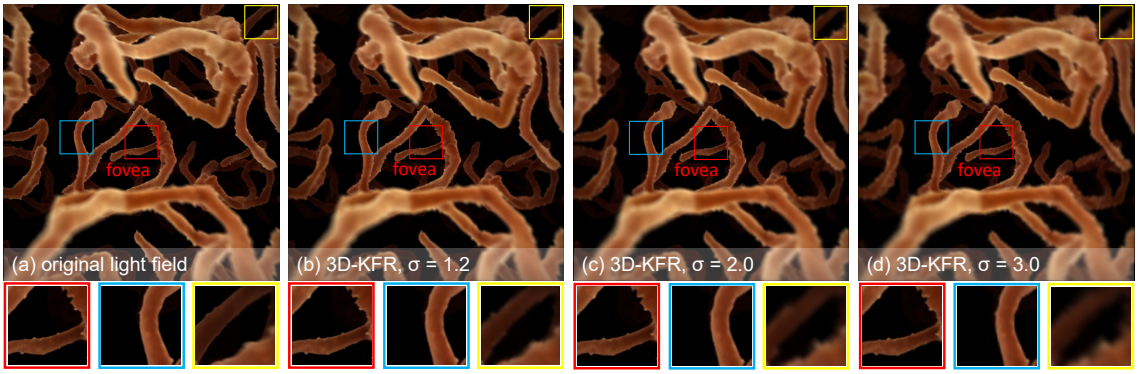


Figure 3.3: The result comparison of the foveated light field with fovea on the center of the screen. (b) - (d) are the application of 3D-KFR on light field with (b) $\sigma_0 = 1.2$, (c) $\sigma_0 = 2.0$, (d) $\sigma_0 = 3.0$. The left zoomed-in views show that the application of 3D-KFR doesn't make changes in the fovea; the middle zoomed-in views and the right zoomed-in views show that larger σ_0 causes detail loss in the peripheral region.

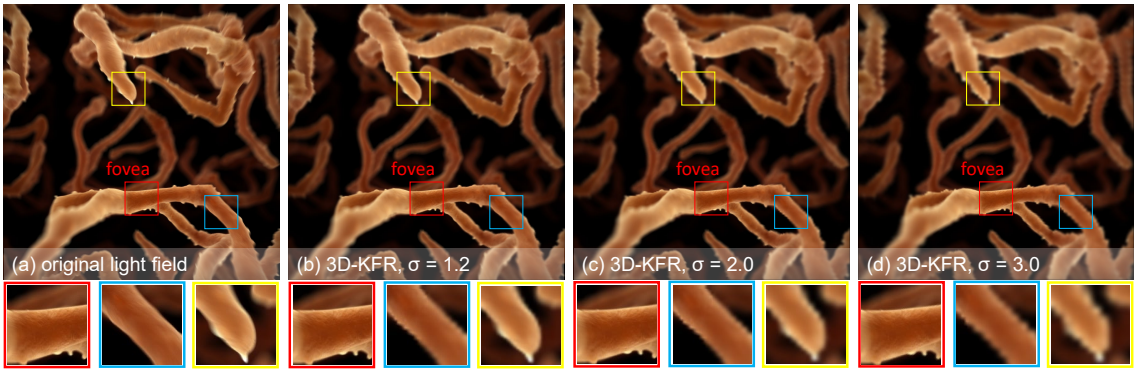


Figure 3.4: The result comparison of the foveated light field with fovea on the peripheral region of the screen. (b) - (d) are the application of 3D-KFR on light field with (b) $\sigma_0 = 1.2$, (c) $\sigma_0 = 2.0$, (d) $\sigma_0 = 3.0$. The left zoomed-in views show that the application of 3D-KFR doesn't make changes in the fovea; the middle zoomed-in views and the right zoomed-in views show that larger σ_0 causes detail loss in the peripheral region.

We partition the original dataset into multiple progressive regions: the inner *foveal region* (highlighted in dark green) indicates the fovea, *i.e.*, where the user is currently looking at; as d increases, the *peripheral regions* (highlighted in lighter green and white) are rendered in smaller framebuffers with less texture sampling.

We classify the frame of the i-th row and j-th column I_{ij} to *foveal region* or *peripheral region* with different framebuffers by d as shown in Equation 3.13.

$$I_{ij} \in \begin{cases} \textit{foveal region} & d < r_0 \\ \textit{peripheral region 1} & r_0 \leq d < r_1 \\ \textit{peripheral region 2} & r_1 \leq d < r_2 \\ \ldots & \ldots \\ \textit{peripheral region N} & r_{N-1} \leq d < r_N \end{cases} \tag{3.13}$$

In the first pass, assume the *foveal region* covers k_0 frames and the $i-th$ peripheral region *peripheral region i* covers k_i frames. Our approach reduces the framebuffer size for the *foveal region* by σ_0^2, and reduces the framebuffer size of *peripheral region i* by σ_1^2,..., σ_N^2, respectively. Then the number of total texture samples in the first pass $N_{\text{3D-KFR pass 1}}$ can be theoretically inferred as:

$$\begin{aligned} N_{\text{3D-KFR pass 1}} &= k_0 \cdot WH \cdot \frac{1}{\sigma_0^2} + \ k_1 \cdot WH \cdot \frac{1}{\sigma_1^2} + \ldots + \ k_N \cdot WH \cdot \frac{1}{\sigma_N^2} \\ &= \left(\frac{k_0}{\sigma_0^2} + \frac{k_1}{\sigma_1^2} + \ldots + \frac{k_N}{\sigma_N^2} \right) \cdot WH \end{aligned} \tag{3.14}$$

We can also write N_{original} as:

$$N_{\text{original}} = k^2 \cdot WH = (k_0 + k_1 + \ldots + k_N) \cdot WH \tag{3.15}$$

So the total number for texture sampling for the *foveal region* and *peripheral region* are reduced by $\frac{1}{\sigma_0^2}\times$, $\frac{1}{\sigma_1^2}\times$, ..., and $\frac{1}{\sigma_N^2}\times$, respectively.

We choose smaller σ with small d in order to keep more details. And we choose larger σ for frames with larger distance in order to reduce rendering cost (*i.e.*, $\sigma_0 \leq \sigma_1 \leq ... \leq \sigma_n$).

The algorithm of light field rendering combined with kernel log-polar transformation is shown in Algorithm 3.

Algorithm 3 3D-KFR: Pass 1

Input:

Aperture size: a,

focal point ratio: f,

fovea coordinate in screen space: X_{fovea} $(\mathring{x}, \mathring{y})$,

pixel coordinate in LP-Buffer: X_{buffer} (u, v),

$k \times k$ light field $\{I\}$ with image resolution of $n \times n$.

Output:

Pixel value $C_{\text{buffer},\ \sigma}$ for the coordinate X_{buffer}.

1: acquire the coordinate for the center camera $X_{\text{cam }0}$

2: acquire the coordinate for the fovea X_{fovea}

3: initialization: $C_{\text{buffer},\ \sigma} \leftarrow \mathbf{0}, N_{\text{buffer},\ \sigma} \leftarrow 0$

4: **for** row index $i \in [0, k]$ **do**

5: **for** column index $j \in [0, k]$ **do**

6: calculate σ with X_{fovea} with Equation 3.13

7: update L with X_{fovea} with Equation 3.5

let $A = \frac{L}{w}$, $B = \frac{2\pi}{h}$

acquire $X_{\text{cam } ij}$ for frame I_{ij}

$d_{ij} \leftarrow \|X_{\text{cam } 0} - X_{\text{cam } ij}\|_2$

if $d_{ij} < a$ **then**

$x' \leftarrow \exp\left(A \cdot \mathbf{K}\left(u, \sigma\right)\right) \cdot \cos\left(Bv\right) + \mathring{x}$

$y' \leftarrow \exp\left(A \cdot \mathbf{K}\left(v, \sigma\right)\right) \cdot \sin\left(Bv\right) + \mathring{y}$

$X'_{\text{Sample}} \leftarrow (x', y')$

$X_{\text{Sample}} \leftarrow X_{\text{cam } ij} + (X'_{\text{Sample}} - X_{\text{cam } ij}) \cdot f$

if X_{Sample} in the range of the screen **then**

$C_{\text{buffer, } \sigma} \leftarrow C_{\text{buffer, } \sigma} + I_{ij} \cdot Color(X_{\text{Sample}})$

$N_{\text{buffer, } \sigma} \leftarrow N_{\text{buffer, } \sigma} + 1$

end if

end if

end for

end for

return $C_{\text{buffer, } \sigma} \leftarrow \frac{C_{\text{buffer, } \sigma}}{N_{\text{buffer, } \sigma}}$

In the second pass, we carry out the inverse-log-polar transformation with anti-aliasing to map the reduced-resolution rendering to the full-resolution screen, the algorithm is shown in Algorithm 4. To reduce artifacts in the peripheral regions, we use a Gaussian filter with a 5×5 kernel for the peripheral region of the recovered image in the screen.

The number of texture samples for the second pass $N_{\text{3D-KFR pass 2}}$ is,

$$N_{\text{3D-KFR pass 2}} = (1 + N) \cdot WH \tag{3.16}$$

The total number of texture samples for rendering the light field with 3D-KFR is

$$\begin{aligned} N_{\text{3D-KFR}} &= N_{\text{3D-KFR pass 1}} + N_{\text{3D-KFR pass 2}} \\ &= \left(\frac{k_0}{\sigma_0^2} + \frac{k_1}{\sigma_1^2} + ... + \frac{k_N}{\sigma_N^2} + 1 + N \right) \cdot WH \end{aligned} \tag{3.17}$$

In the light field rendering, we commonly have $k \geq 16$. In 3D-KFR, we commonly have $1.0 < \sigma \leq 3.0$, and we choose $N = 2$ as the number of peripheral regions and $\mathbf{K}(x) = x^4$ as the kernel function. Therefore, we have

$$N_{\text{KFR pass 2}} = WH \ll \frac{k^2}{\sigma^2} \cdot WH = N_{\text{KFR pass 1}}, \tag{3.18}$$

and

$$\begin{aligned} N_{\text{3D-KFR pass 2}} &= (N + 1)WH \\ &\ll \left(\frac{k_0}{\sigma_0^2} + \frac{k_1}{\sigma_1^2} + ... + \frac{k_N}{\sigma_N^2} \right) \cdot WH = N_{\text{3D-KFR pass 1}}. \end{aligned} \tag{3.19}$$

Equations 3.18 and 3.19 show that the extra time consumed by the *pass 2* can be omitted. We then have

$$N_{\text{KFR}} \approx (\frac{k^2}{\sigma_0^2}) \cdot WH \tag{3.20}$$

$$N_{\text{3D-KFR}} \approx \left(\frac{k_0}{\sigma_0^2} + \frac{k_1}{\sigma_1^2} + ... + \frac{k_N}{\sigma_N^2} \right) \cdot WH \tag{3.21}$$

Comparing Equations 3.1, 3.20 and 3.21, we have

$$N_{\text{3D-KFR}} \ll N_{\text{KFR}} \ll N_{\text{original}}, \tag{3.22}$$

which shows that the 3D KFR scheme can accelerate the rendering of the light field beyond a simple KFR approach. The resulting comparison of the original light field rendering and the 3D-KFR for light field is shown in Figure 3.3 and Figure 3.4. With 3D-KFR applied, pixel density decreases from the fovea to the periphery. We do not notice any differences in the fovea with different σ_0 between the left zoomed-in views because 3D-KFR uses a weighted-sum which strengthens the frames with small d. For the same reason, we can notice the loss of detail from the right zoomed-in views of the periphery. Next, we determine what parameters ensure that the peripheral loss and the peripheral blur are not noticeable by conducting user studies.

Algorithm 4 3D-KFR: Pass 2

Input:

Fovea coordinate in screen space: $X_{\text{fovea}}\ (\mathring{x}, \mathring{y})$,

pixel coordinate in screen space: $X_{\text{Display}}\ (x, y)$,

Output:

Pixel value C_{display} for coordinate X_{Display}.

1: initialization: $C_{\text{display}} \leftarrow \mathbf{0}, N_{\text{display}} \leftarrow 0$

2: acquire the coordinate for the fovea X_{fovea}

3: update L with X_{fovea} with Equation 3.5

4: **for** each attachment I_σ in LP-Buffer **do**

5: $\quad x' \leftarrow x - \mathring{x}$

6: $\quad y' \leftarrow y - \mathring{y}$

7: $\quad u \leftarrow \mathbf{K^{-1}} \left(\frac{\log\|x', y'\|}{L}\right) \cdot w$

8: $\quad v \leftarrow \arctan\left(\frac{y'}{x'}\right) \frac{h}{2\pi} + \mathbf{1}\left[y' < 0\right] \cdot h$

9: $X_{\text{Sample}} \leftarrow (u, v)$

10: $C_{\text{display}} \leftarrow C_{\text{display}} + I_{\sigma} \cdot Color(X_{\text{Sample}})$

11: $N_{\text{display}} \leftarrow N_{\text{display}} + 1$

12: **end for**

13: return $C_{\text{display}} \leftarrow \frac{C_{\text{display}}}{N_{\text{display}}}$

3.4 User Study

We have carried out user studies to find the largest foveation parameter values for σ_0 that results in visually acceptable foveated rendering.

3.4.1 Apparatus

Our user study apparatus is shown in Figure 3.5. We used an *Alienware* laptop with an NVIDIA GTX 1080, a *FOVE* HMD, and an *XBOX* controller. The *FOVE* display has a 100° field of view, a resolution of 2560 × 1440, and a 120 Hz infrared eye-tracking system with a precision of 1° and a latency of 14 ms.

Since public datasets on large-scale and high-resolution microscopy light field datasets are not yet available, we have rendered and open sourced 30 microscopy light field datasets[1]. We synthesized the microscopy dataset on *Cell* and *Cellular Lattice* for the user study.

[1]Simulated HD Light Fields: http://users.umiacs.umd.edu/~xmeng525/3D_KFR_For_Light_Fields/MicroscopyLightFieldResource/

Figure 3.5: Our user study set up with gaze-tracker integrated into the *FOVE* head-mounted display.

3.4.2 Participants

We recruited a total of 22 participants via campus email lists and flyers. All participants are at least 18 years old with normal or corrected-to-normal vision (with contact lenses).

3.4.3 Procedure

We conducted three different and independent experiments to test the parameters for which 3D-KFR produces acceptable quality to non-foveated rendering: a **Pair Test**, a **Random Test**, and a **Slider test**.

In the **Pair Test**, we presented each participant with pairs of foveated and full-resolution light field renderings. We presented the two renderings in each pair in a random order and separated by a short interval of black screen (0.75 seconds). The foveation parameter ranged between $\sigma_0 = 1.2$ to $\sigma_0 = 3.0$. Pairs at all quality levels in this range were presented twice (monotone increasing then monotone decreasing) for each dataset, i.e. σ_0 increased from 1.2 to 3.0 then decreased from 3.0 to 1.2. At

the end of each comparison, the participant responded upon the similarity between the two rendering results by the *XBOX* controller. The answer contains 5 confidence levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual difference*, 3 represents *acceptable perceptual difference*, 2 represents *noticeable perceptual difference* and 1 represents *significant perceptual difference*.

In the **Random Test**, we presented each participant with pairs of foveated and full-resolution light field renderings. We presented the two renderings in each pair in a random order and separated by a short interval of black (0.75 seconds). The foveation parameter ranged between $\sigma_0 = 1.2$ to $\sigma_0 = 3.0$. Pairs at all quality levels in the range were presented once for each dataset in random order. At the end of each comparison, the participant responded upon the similarity between the two rendering results by the *XBOX* controller. The answer contains 5 confidence levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual difference*, 3 represents *acceptable perceptual difference*, 2 represents *noticeable perceptual difference* and 1 represents *significant perceptual difference*.

The **Slider Test** lets the participants navigate the foveation quality space themselves. First, the participant observed the full-resolution rendering result as a reference. Next, we presented the participant with the lowest level of foveation quality ($\sigma_0 = 3.0$) while the participant could progressively increase the foveation level (with a step size of 0.1). The participant switched between the foveated rendering result and the reference image back and forth, until they found out the lowest foveation level which is visually equivalent to the non-foveated reference. We recorded the first quality level index at which the participant stopped as the final response for

the slider test.

To ensure the visual attentiveness of the participants, we randomly inserted 30% of the trials to be "validation trials" that had identical full-resolution results for both choices in the **Pair Test** and the **Random Test**. If the participant declared these validation renderings to be *mostly the same with acceptable difference*, *noticeable difference* or *totally different*, we would ask the participant to stop, take some rest, and then continue. Meanwhile, we recorded this choice as an *error*. If $error \geq 5$ in the **Pair Test** and the **Random Test**, we would stop the user study and discard the data of the user. We discarded two participants according to this rule.

3.5 Results and Acceleration

3.5.1 Results of the User Study

Let S_σ be the average score of all the users for a specific σ_0, and let P_σ be the percentage of responses of rated foveated and non-foveated renderings as *perceptually identical (5)* and *minimal perceptual difference (4)* for a specific $\sigma = \sigma_0$.

The result of S_σ for the **Pair Test** is shown in Figure 3.6. Generally, S_σ decreases with the increase of σ_0. A Friedman test revealed a significant effect of the users' responses on foveation parameter σ ($\chi^2(20) = 104.3, p < 8.9 \times 10^{-14}$). The result of P_σ for the **Pair Test** is shown in Figure 3.7. We have identified a threshold of $P_\sigma = 90\%$ for $\sigma_{pair} = 2.4$ as the foveation parameter that provides minimal perceptual differences based on the **Pair Test**.

The result of S_σ for the **Random Test** is shown in Figure 3.8. The trend

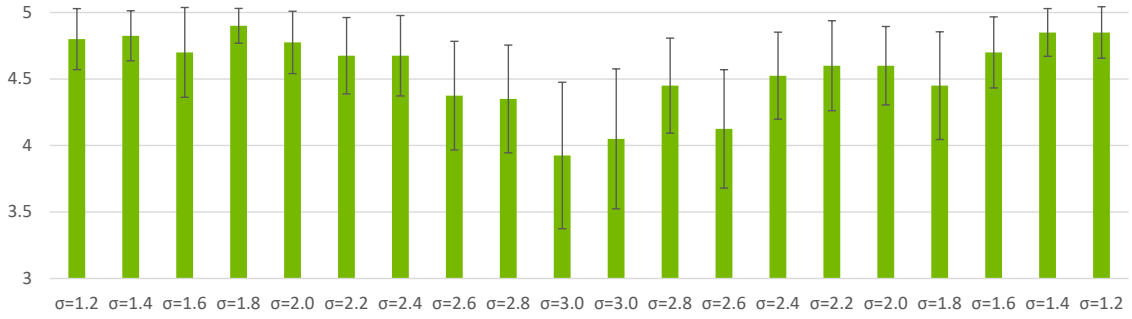


Figure 3.6: The **Pair Test** responses of S_σ across sliding foveation parameters σ. S_σ decreases with the increase of σ. 5 represents perceptually identical, 4 represents minimal perceptual difference, 3 represents acceptable perceptual difference, 2 represents noticeable perceptual difference, and 1 represents significant perceptual difference (2 and 1 are not shown)

.

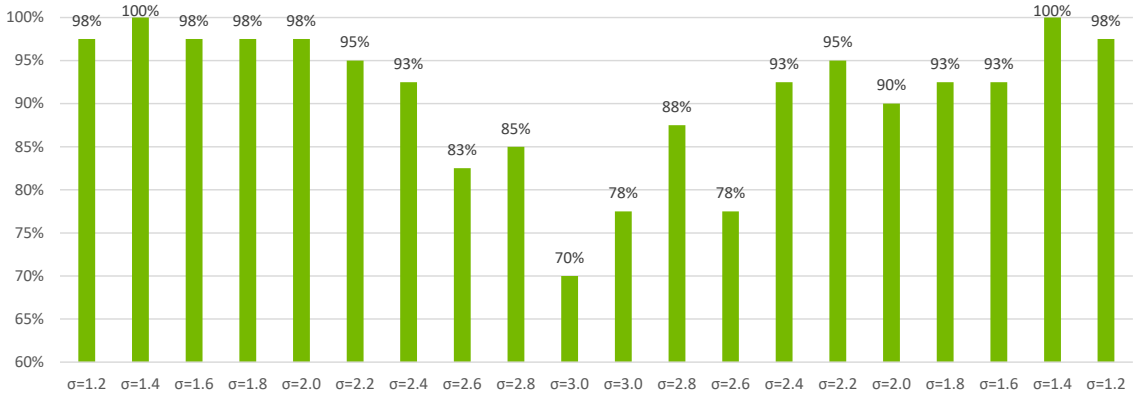


Figure 3.7: The **Pair Test** responses of P_σ across sliding foveation parameters σ. P_σ decreases with the increase of σ.

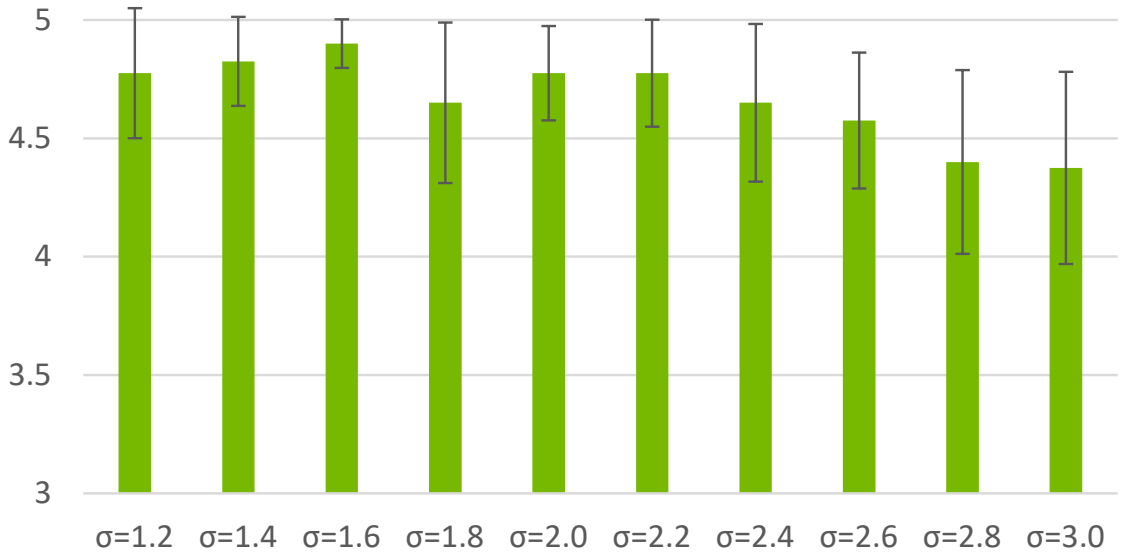


Figure 3.8: The **Random Test** responses of S_σ across gradually varied foveation parameters σ. S_σ decreases with the increase of σ. 5 represents perceptually identical, 4 represents minimal perceptual difference, 3 represents acceptable perceptual difference, 2 represents noticeable perceptual difference, and 1 represents significant perceptual difference (2 and 1 are not shown)

.

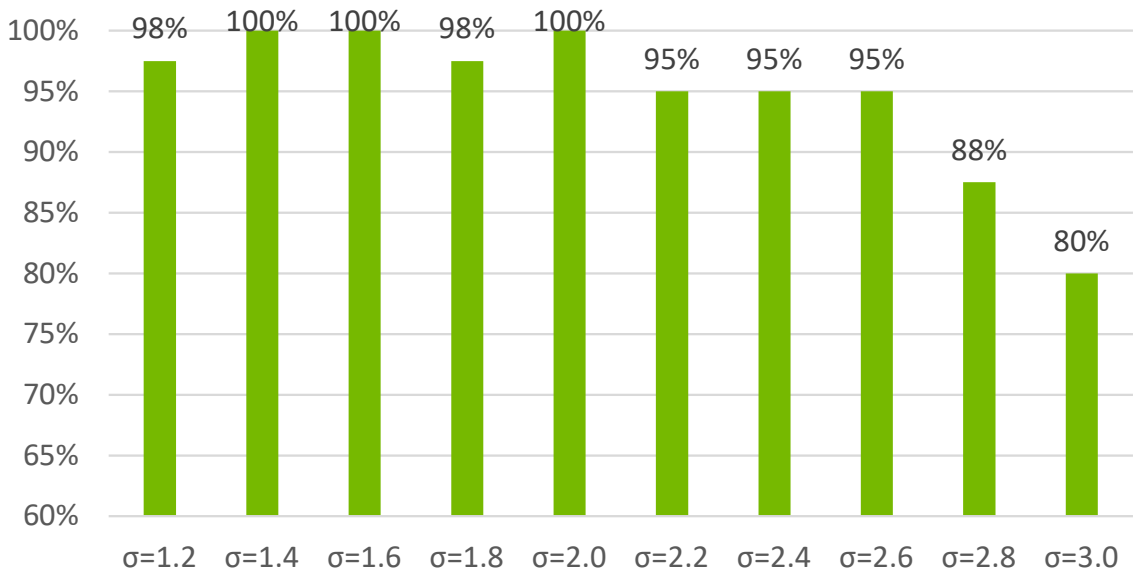


Figure 3.9: The **Random Test** responses of P_σ across sliding foveation parameters σ. P_σ decreases with the increase of σ.

that S_σ decreases with an increase of σ matches our expectation. A Friedman test revealed a significant effect of the users' responses on foveation parameter σ ($\chi^2(20) = 29.2, p < 0.0006$). The result of P_σ for the **Random Test** is shown in Figure 3.9. We have identified a threshold of $P_\sigma = 90\%$ for $\sigma_{random} = 2.6$ as the foveation parameter that provides minimal perceptual differences based on the **Random Test**.

The histogram of the user-chosen thresholds in the **Slider Test** is shown in Figure 3.10. For instance, the histogram shows that 25% of the users found that $\sigma = 3.0$ or lower is acceptable; 75% of the users found that $\sigma = 1.8$ or lower is acceptable. With $\sigma_0 = 1.6$, 80% of the users considered that the foveated rendering is visually indistinguishable from full-resolution rendering. We chose threshold

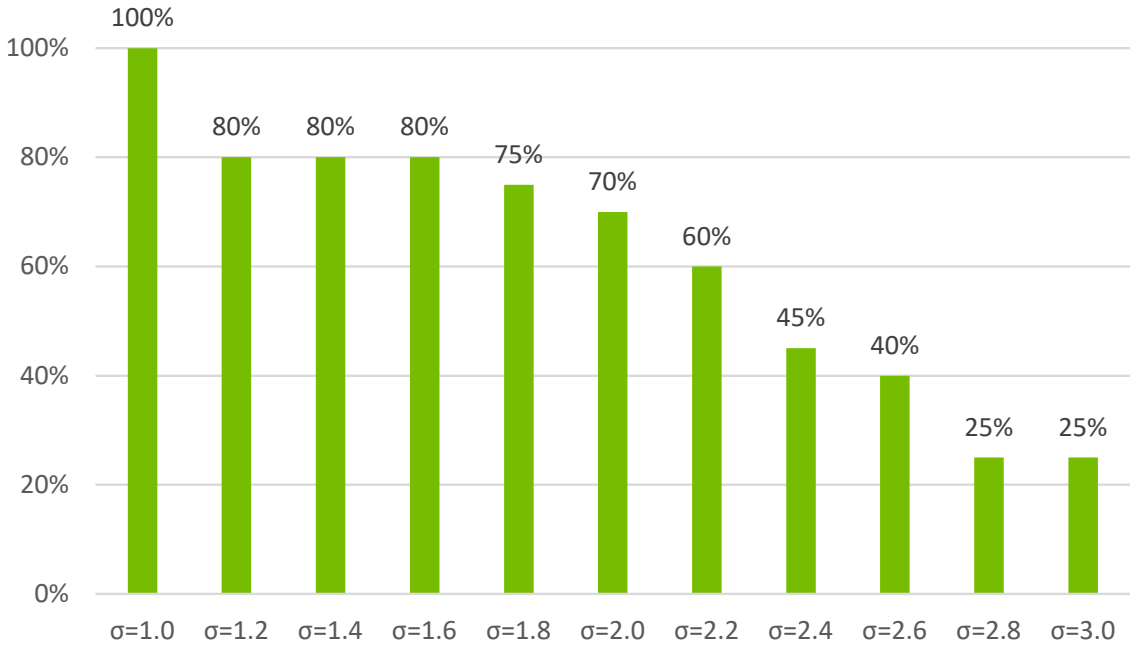


Figure 3.10: The histogram of the optimal foveation parameter σ selected by each user in the **Slider Test**. For instance, the histogram shows that 80% of the users found that $\sigma = 1.6$ or lower is acceptable.

$\sigma_{slider} = 1.6$.

We have noticed that $\sigma_{slider} = 1.6$ is smaller than $\sigma_{pair} = 2.4$ and $\sigma_{random} = 2.6$. We speculate that the reason for a smaller sigma in the **Slider Test** is: if the users are free to choose the threshold, they tend to choose the best quality they can achieve, instead of the lower bound of the perceptually indistinguishable quality.

3.5.2 Rendering Acceleration

Using the three parameters, one could think of building a foveated rendering system where the saccades are implemented with $\sigma = 2.6$ and the fixation implemented with $\sigma = 1.6$.

Performance Evaluation and Discussion We have implemented the 3D kernel foveated rendering pipeline in C++ 11 and OpenGL 4 on NVIDIA GTX 1080. We report the results of our rendering acceleration for the *tissue* dataset at the resolution of $k \times k \times 1024 \times 1024$. We tested the rendering time for different light field dimensions ($20 \leq k \leq 25$) and different σ_0 ($1.2 \leq \sigma_0 \leq 3.0$) with $\sigma_1 = 1.6\sigma_0$, $\sigma_2 = 2.0\sigma_1$. We used the kernel function $\mathbf{K}(x) = x^4$. The evaluations are shown in Figure 3.11, where $\sigma_0 = 1.0$ corresponds to the rendering time of the original field, and $\sigma_0 > 1.0$ corresponds to the 3D-KFR rendering time. We notice that the rendering time of 3D-KFR decreases with the increase of σ_0.

We further tested the rendering time comparison and the speedup for $\sigma_{slider} = 1.6$, $\sigma_{pair} = 2.4$ and $\sigma_{random} = 2.6$ as shown in Table 3.1. With $\sigma_{pair} = 2.4$, the rendering time is less than 21.96 ms (45.54 fps); with $\sigma_{random} = 2.6$, the rendering

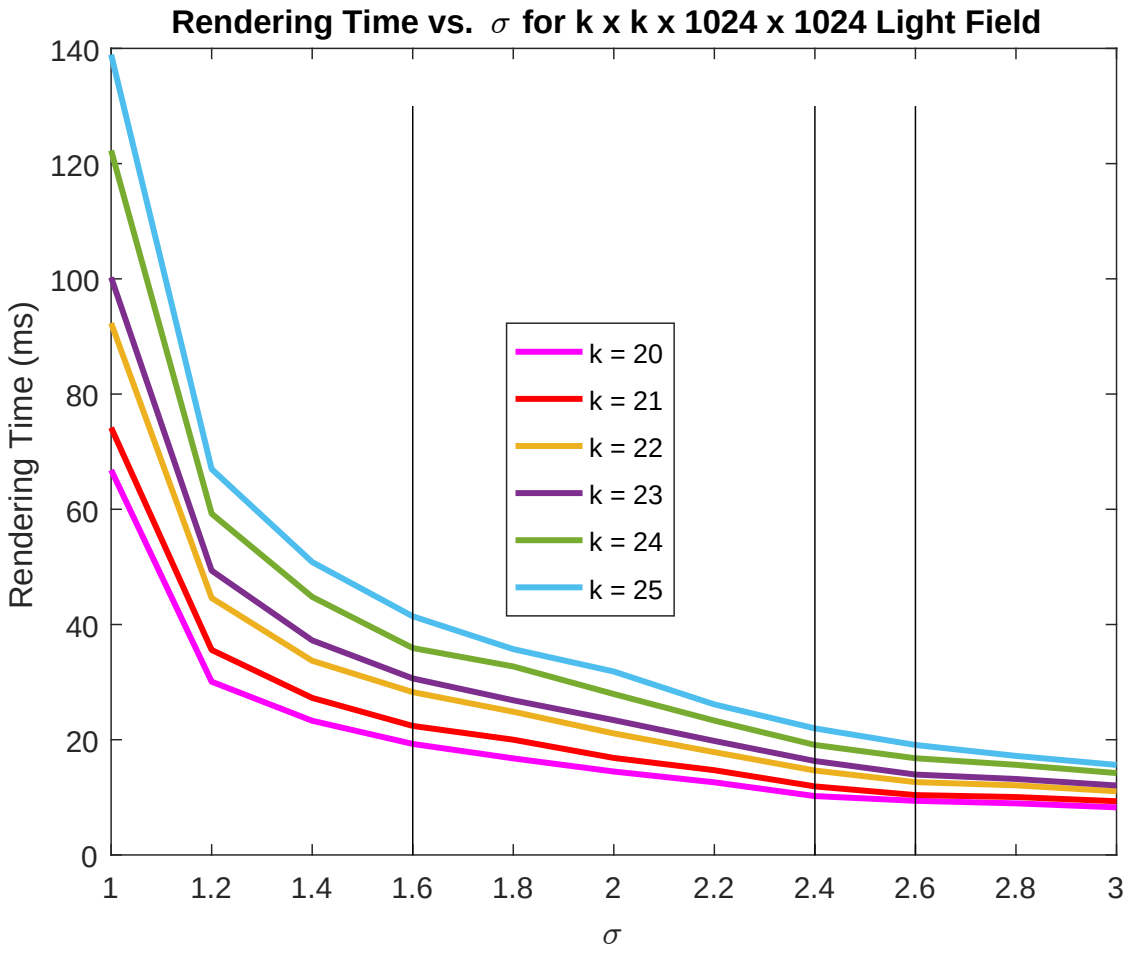


Figure 3.11: The rendering time for light field with different dimension and different σ.

Resolution	Ground Truth	$\sigma = 1.6$		$\sigma = 2.4$		$\sigma = 2.6$	
		3D KFR	3D-KFR Speedup	3D KFR	3D-KFR Speedup	3D KFR	3D-KFR Speedup
$20 \times 20 \times 1024 \times 1024$	66.83 ms	19.27 ms	3.47×	10.22 ms	6.54×	9.39 ms	7.11×
$21 \times 21 \times 1024 \times 1024$	74.17 ms	22.39 ms	3.31×	11.90 ms	6.24×	10.39 ms	7.14×
$22 \times 22 \times 1024 \times 1024$	92.33 ms	28.26 ms	3.27×	14.65 ms	6.30×	12.64 ms	7.30×
$23 \times 23 \times 1024 \times 1024$	100.26 ms	30.64 ms	3.27×	16.30 ms	6.15×	13.95 ms	7.18×
$24 \times 24 \times 1024 \times 1024$	122.29 ms	35.92 ms	3.40×	19.09 ms	6.41×	16.79 ms	7.28×
$25 \times 25 \times 1024 \times 1024$	138.93 ms	41.42 ms	3.35×	21.96 ms	6.33×	19.09 ms	7.28×

Table 3.1: The average timings and the corresponding speedups of 3D-KFR at different light field dimensions and foveation parameters σ.

time is less than 19.09 ms (52.38 fps). $\sigma_{pair} = 2.4$ and $\sigma_{random} = 2.6$ meets the real-time requirement of 30 fps. With $\sigma_{slider} = 1.6$, the rendering times for $k = 20, 21, 22,$ or 23 are less than 30.64 ms (32.64 fps), which meets the real-time requirement of 30 fps. While the rendering times for $k = 24$ and 25 are less than 41.42 ms (24.14 fps), they are still able to achieve reasonably interactive frame rates.

3.5.3 Quality Evaluation

The comparisons of the visualization of the original light field rendering and the 3D-KFR rendering of different datasets are shown in Figure 3.12 - Figure 3.14.

We use structural dissimilarity (DSSIM) [84] [85] between the 3D-KFR and the original light field approaches as the metric to evaluate the quality of 3D-KFR results. DSSIM can be derived from structural similarity index (SSIM) [86] [87]. The measurement of SSIM and DSSIM between the two images Ψ and Ω with size $N \times N$ is shown in Equations 3.23 and 3.24.

$$SSIM(\Psi, \Omega) = \frac{(2\mu_\Psi \mu_\Omega + c_1)(2\sigma_{\Psi\Omega} + c_2)}{(\mu_\Psi^2 + \mu_\Omega^2 + c_1)(\sigma_\Psi^2 + \sigma_\Omega^2 + c_2)} \tag{3.23}$$

$$DSSIM(\Psi, \Omega) = \frac{1 - SSIM(\Psi, \Omega)}{2} \tag{3.24}$$

where μ_Ψ and μ_Ω are the average pixel values for images Ψ and Ω, respectively; σ_Ψ and σ_Ω are the pixel variances for images Ψ and Ω, respectively; $\sigma_{\Psi\Omega}$ is the covariance between images Ψ and Ω; c_1, c_2 are two constants used to stabilize the division with a weak denominator.

SSIM is a perception-based model that considers image degradation as perceived

change in structural information. A pair of images with low DSSIM indicates better structural similarity. We measure the average DSSIM of the RGB channels, and we show the results in Figure 3.12 - Figure 3.14. The DSSIM measurement of the zoomed-in views for the fovea regions are small, which indicates high visual similarity. With an increase in distance between the fovea position and the pixel position, the DSSIM increases because of the foveation effect.

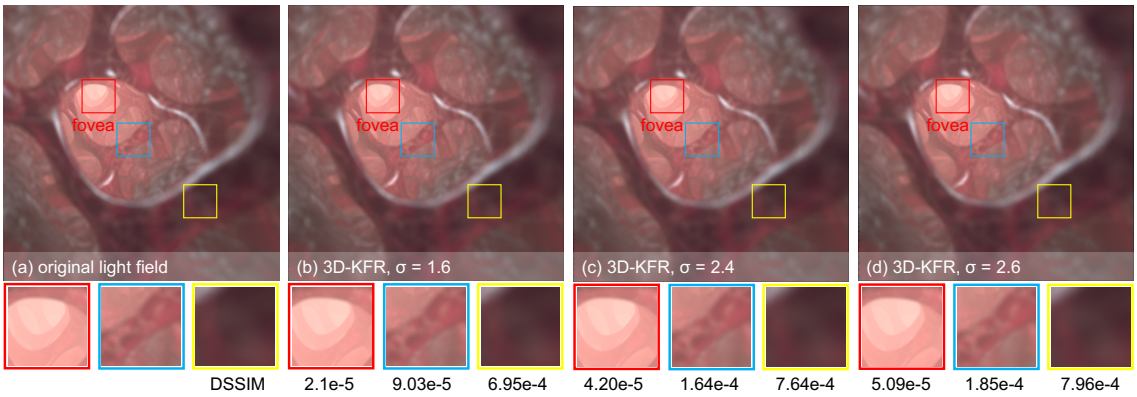


Figure 3.12: Comparison of the foveated light field *Biomine II*. (b) - (d) using 3D-KFR with (b) $\sigma_{slider} = 1.6$, (c) $\sigma_{pair} = 2.4$, (d) $\sigma_{random} = 2.6$. The measured DSSIM (lower is better) is shown for each zoomed-in view.

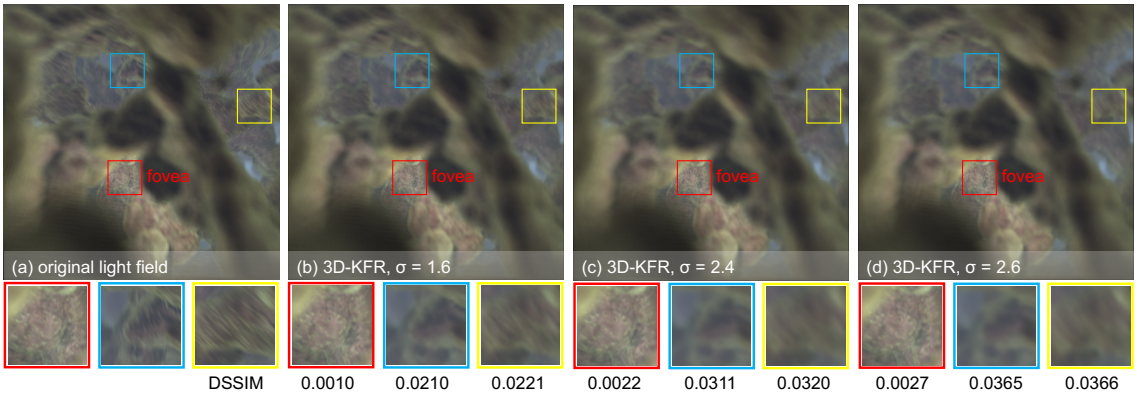


Figure 3.13: Comparison of the foveated light field *Cellular Lattice IV*. (b) - (d) using 3D-KFR with (b) $\sigma_{slider} = 1.6$, (c) $\sigma_{pair} = 2.4$, (d) $\sigma_{random} = 2.6$. The measured DSSIM (lower is better) is shown for each zoomed-in view.

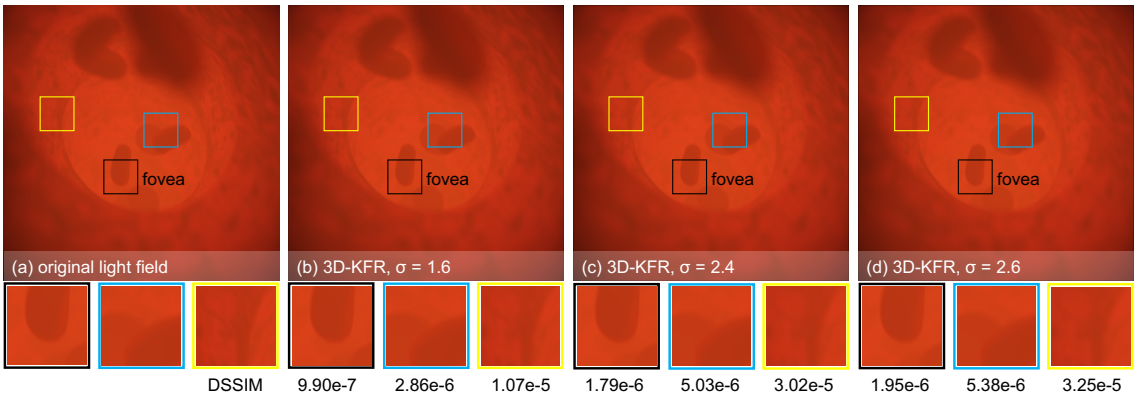


Figure 3.14: Comparison of the foveated light field *Red Cells IV*. (b) - (d) using 3D-KFR with (b) $\sigma_{slider} = 1.6$, (c) $\sigma_{pair} = 2.4$, (d) $\sigma_{random} = 2.6$. The measured DSSIM (lower is better) is shown for each zoomed-in view.

Chapter 4: Eye-dominance-guided Foveated Rendering

4.1 Overview

As we have seen in Chapter 2, foveation speeds up the rendering of each frame by 3× to 5× [6, 13, 23]. Other rendering acceleration approaches take advantage of the properties of the human visual system, such as perception-guided reduction of motion artifacts [88, 89] and temporal resolution multiplexing [22], which renders even-numbered frames at a lower resolution.

The human visual system has a tendency to prefer visual stimuli of one eye over the other eye [90]. This phenomenon is referred to as eye (or ocular) dominance. The dominant eye is found to be superior to the non-dominant eye in visual acuity, contrast sensitivity [91], color discrimination [92], and motor functions that are visually managed and require spatial attention [93].

In this chapter, we propose the technique of eye-dominance-guided foveated rendering (EFR), which leverages ocular dominance property of the human visual system. We render the scene for the dominant eye at the normal foveation level and render the scene for the non-dominant eye at a higher foveation level. This formulation allows us to save more in the rendering budget for the non-dominant eye. We have validated our approach by carrying out quantitative experiments and

user studies. Our contributions include:

1. designing eye-dominance-guided foveated rendering, an optimized technique for foveated rendering, that provides similar visual results as the original foveated rendering, but at a higher rendering frame rate;

2. conducting user studies to identify the parameters for the dominant eye and the non-dominant eye to maximize perceptual realism and minimize computation for foveated rendering in head-mounted displays; and,

3. implementing eye-dominance-guided foveated rendering pipeline on a GPU, and achieving up to 1.47× speedup over the original foveated rendering at a resolution of 1280×1440 per eye with minimal perceptual loss of detail.

4.2 Related Work

The related work in foveated rendering as been reviewed in Section 2.2. In this section, we review the relevant state-of-the-art research eye-dominance that inspires our work. Eye dominance has been described as the inherent tendency of the human visual system to prefer scene perception from one eye over the other [90].

Einat and Shaul [91] study the role of eye dominance in dichoptic non-rivalry conditions, testing visual search and comparing performance with target presented to the dominant or the non-dominant eye. Einat and Shaul [91] designed user study with red–green glasses. The participants viewed an array of green and red lines of uniform orientation, with a differently oriented target line present on half the trials. And they observed the performance of visual perception. They found that the

dominant eye significantly performed better than the non-dominant eye when the dominant eye saw the target, especially when the opposite eye saw the distractors. This effect was reduced when only nearest-neighbor surrounding distractors were homogeneous. They conclude that the dominant eye has priority in visual processing, perhaps even resulting in inhibition of non-dominant eye representations.

Oishi *et al.* [94] observe that the dominant eye is functionally activated prior to the non-dominant eye in conjugate eye movements. They recorded conjugate eye movements to elucidate whether ocular dominancy was present at reading distance in 21 right-handed normal participants by using a video-oculographic measurement. This included the velocity of smooth pursuits, and the latency and velocity of saccades. They defined the dominant eye for each participant by the near–far alignment test and 20 subjects showed the right dominant eyes. Although the ocular dominancy was not found in the velocity of smooth pursuit and vertical saccades, the velocity of horizontal saccades in the dominant eyes was faster than that in the non-dominant eyes. These results suggest that the dominant eye is functionally activated prior to non-dominant eye in horizontal saccades at reading distance, which thus indicates the functional dominance of the dominant eye in conjugate eye movements.

Koctekin *et al.* [92] find that the dominant eye has priority in red/green color spectral region, leading to better color-vision discrimination ability. For this comparative study, 50 students studying at Başkent University Faculty of Medicine, including 31 males (62%) and 19 females (38%), with visual acuity of 20/20 and without congenital color vision deficiency (CCVD) evaluated by Ishihara pseudoisochromatic plate test (IPPT) were recruited. Dominant eye was determined

by the *Gundogan* Method. The color discrimination ability was examined with the $Farnsworth - Munsell$ 100 hue (FM100) test. The statistical differences among the dominant eye and the non-dominant eye in red/green local region and total error scores were found to be statistically significant in both genders.

McManus *et al.* [93] studied the relationship between handedness and eye-dominance. Handedness and eye-dominance are associated statistically, although a previous meta-analysis has found that the precise relationship is difficult to explain, with about 35% of right-handers and 57% of left-handers being left eye dominant. Of particular difficulty to genetic or other models is that the proportions are distributed asymmetrically around 50%. This study explored whether this complicated pattern of association occurred because it divides right-and left-handers into consistent handers (who write and throw with the same hand) and inconsistent handers (who write and throw with opposite hands). In an analysis of 10,635 participants from questionnaire studies, 28.8% of left-handers and 1.6% of right-handers by the writing task were found to be inconsistent for the throwing task. The results also showed that writing hand and throwing hand both relate independently to eyedness and that throwing hand is somewhat more strongly associated with eyedness. The study found that 24.2% of consistent right-handers are left eye dominant compared with 72.3% of consistent left-handers, and 55.4% of inconsistent right-handers compared with 47.0% of inconsistent left-handers. They conclude that eyedness is phenotypically secondary to writing and throwing handedness. In the discussion they note that eyedness runs in families. They present new data suggesting that writing hand and throwing hand are co-inherited, and they argue that further data are now required to model properly

the associations of writing hand, throwing hand, and eyedness, as well as probably also footedness and language dominance.

Chaumillon *et al.* [95] show that sighting eye dominance has an influence on visually triggered manual action with shorter reaction time. They used the simple and well-known Poffenberger paradigm [96] in which participants press a button with the right or left index finger, in reaction to the appearance of a lateralized visual stimulus. By selecting participants according to their dominant-eye and handedness, they deciphered the impact of eye dominance on visuomotor transformation speed. They showed that, in right-handers simple reaction times (RT) in response to a lateralized visual target are shorter when it appears in the contralateral visual hemifield with respect to the dominant eye. Meanwhile, in left-handers, only those with a right dominant eye exhibit a shorter RT with the left hand and they show no hemifield difference. Additionally, the Poffenberger paradigm has been used to estimate the interhemispheric transfer time (IHTT) in both directions, from the right to the left hemisphere or the reverse, by comparing hand RTs following stimulation of each visual hemifield. Chaumillon *et al.* [95] demonstrates that this paradigm leads to biased estimations of these directionally considered IHTT and provides an explanation to the often reported IHTT negative values that otherwise appear implausible. These new findings highlight the need to consider eye dominance in studies investigating the neural processes underlying visually-guided actions. More generally, they demonstrate a substantial impact of eye dominance on the neural mechanisms involved in converting visual inputs into motor commands.

In this work, we take advantage of the weaker sensitivity and acuity of the

non-dominant eye and render the non-dominant display with greater foveation to accelerate overall foveated rendering.

4.3 Proposed Algorithm

Here we present an overview of the parameterized foveated rendering and then we build upon it to accomplish eye-dominance-guided foveated rendering.

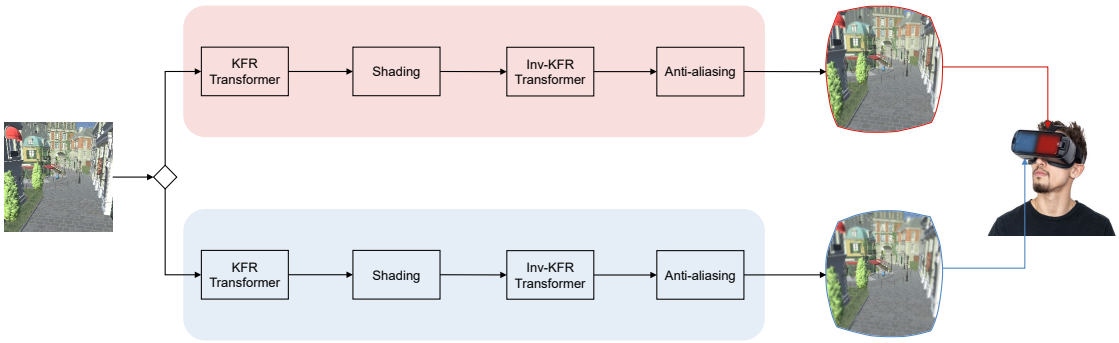


Figure 4.1: An overview of the eye-dominance-guided foveated rendering system. Our system uses two foveated renderers, with different values of the foveation parameter σ, for the dominant eye and the non-dominant eye, respectively. For the dominant eye, we choose the foveation parameter σ_d which results in an acceptable foveation level for both eyes. For the non-dominant eye, we choose $\sigma_{nd} \geq \sigma_d$, which corresponds to a higher foveation level. Because the non-dominant eye is weaker in sensitivity and acuity, the user is unable to notice the difference between the two foveation frames.

4.3.1 Foveation Model

We use the kernel foveated rendering (KFR) model proposed in Section 2 because this model parameterizes the level of foveation with two simple parameters: frame buffer parameter σ controls the width of the frame-buffer to be rendered, thus controlling the level of foveation; and the kernel function parameter α controls the distribution of pixels.

Here I briefly describe the KFR model again. The KFR model contains two passes. In the first pass, the renderer transforms the shading materials in the G-buffer (world positions, texture coordinates, normal maps, albedo maps, etc.) from the Cartesian coordinates to kernel log-polar coordinates. Because of the non-uniform scaling effect in the transformation, details in the foveal region are preserved and details in the peripheral region are reduced.

Given a screen of resolution $W \times H$, for each pixel with coordinate (x, y), we first normalize the coordinate to (x', y'). Then, KFR transforms the point (x', y') to (u, v) in the kernel log-polar space via Equation 2.10, where L is the log of the maximum distance from $\mathbf{F}\,(\mathring{x}, \mathring{y})$ to the farthest screen corner as shown in Equation 2.12.

In the second pass, the renderer transforms the rendered scene from kernel log-polar coordinates to Cartesian coordinates and renders to the full-resolution screen. A pixel with log-polar coordinates (u, v) is transformed back to (x'', y'') in Cartesian coordinates as shown in Equation 2.5.

According to Section 2, the kernel function parameter is suggested as $\alpha = 4$. Therefore, we can control the level of foveation by only altering the parameter σ.

4.3.2 Eye-dominance-guided Foveation Model

Previous research on ocular dominance indicates that the non-dominant eye is weaker than the dominant eye in sensitivity and acuity. Here, we propose that the non-dominant eye is able to accept a higher level of foveation.

The overview of our eye-dominance-guided foveated rendering (EFR) system is shown in Figure 4.1. In our EFR framework, for the baseline rendering, the system uses a KFR renderer with foveation parameter σ_d for the dominant eye and a KFR renderer with σ_{nd} for the non-dominant eye.

In the KFR algorithm, the parameter σ controls the width of the frame-buffer to be rendered, and the rendering time is proportional to the area of the rendered buffer. In other words, rendering time is inversely proportional to σ^2. Suppose the rendering time of the original frame for each eye is T, then the expected rendering time of KFR with $\sigma_d = \sigma_{nd}$ is:

$$t_{FR} = \frac{T}{\sigma_d^2} + \frac{T}{\sigma_{nd}^2} = \frac{2T}{\sigma_d^2} \tag{4.1}$$

The expected rendering time of eye-dominance-guided foveated rendering (EFR) with $\sigma_d \neq \sigma_{nd}$ is:

$$t_{EFR} = \frac{T}{\sigma_d^2} + \frac{T}{\sigma_{nd}^2} = \frac{T}{\sigma_d^2}\left(1 + \left(\frac{\sigma_d}{\sigma_{nd}}\right)^2\right) \tag{4.2}$$

Then,

$$\begin{aligned} & \sigma_d \leq \sigma_{nd} \\ \Rightarrow & \left(\frac{\sigma_d}{\sigma_{nd}}\right)^2 \leq 1 \\ \Rightarrow & \frac{T}{\sigma_d^2}\left(1+\left(\frac{\sigma_d}{\sigma_{nd}}\right)^2\right) \leq \frac{2T}{\sigma_d^2} \\ \Rightarrow & t_{EFR} \leq t_{FR}. \end{aligned} \tag{4.3}$$

Therefore, with $\sigma_d \leq \sigma_{nd}$, the rendering time for head-mounted displays can be reduced with non-perceivable difference between the foveated renderings for the dominant eye and the non-dominant eye.

The theoretical speedup S achieved by EFR is shown in Equation 4.4:

$$S = \frac{t_{FR}}{t_{EFR}} = \frac{2}{1+\left(\frac{\sigma_d}{\sigma_{nd}}\right)^2} \geq 1. \tag{4.4}$$

Next, we conduct user studies to validate that the non-dominant eye is able to accept a higher level of foveation than the dominant eye, and also identify the foveation parameters for the dominant and non-dominant eyes.

4.4 User Study

We have conducted two user studies: a pilot study and a main study to identify the eye-dominance-guided foveated rendering parameters which can produce perceptually indistinguishable results compared with non-foveated rendering.

4.4.1 Apparatus

Our user study apparatus consists of an *Alienware* laptop with an NVIDIA GTX 1080 and a *FOVE* head-mounted display. The FOVE headset is integrated

with a 120 Hz infrared eye-tracking system and a 2560×1440 resolution screen (1280×1440 per eye). We use an *XBOX* controller for the interaction between the participant and the system. User studies took place in a quiet room.

As shown in Figure 4.2, the computer-generated environments consist of two *fireplace room* scenes [18] and 8 scenes from the *Amazon Lumberyard Bistro* [19]. These scenes are rendered with the *Unity* game engine. To ensure that the participants are familiar with the user study system, we requested the participants to complete all the tasks for the trial run and familiarize themselves fully with the interaction before the formal tests.

4.4.2 Pre-experiment: Dominant Eye Identification

In both of the pilot study and the main study, we use the Miles Test [97] to measure the eye dominance for each participant before the start of the study.

First, the participant extends their arms out in front of themselves and creates a triangular opening between their thumbs and forefingers by placing their hands together at a 45-degree angle. Next, with both eyes open, the participant centers the triangular opening on a goal object that is 20 feet away. Then, the participant closes their left eye with their right eye open. Finally, the participant closes their right eye with their left eye open. If the goal object stays centered with the right eye open and is no longer framed by their hands with the left eye open, the right eye is their dominant eye. If the goal object stays centered with the left eye open and is no longer framed by their hands with the right eye open, the left eye is their

dominant eye. The Miles Test is performed twice for each participant, and we record the participant's dominant eye and configure our renderer accordingly.

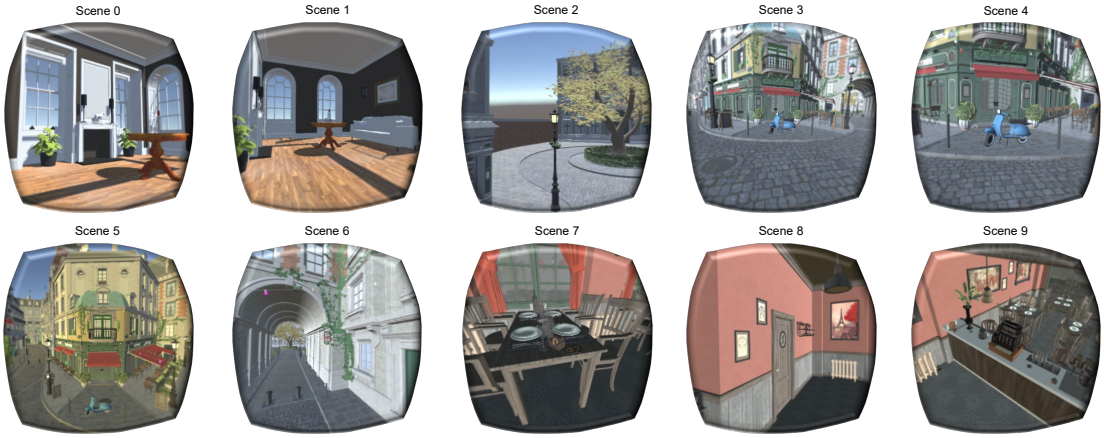


Figure 4.2: The scenes used for the user study. Scene 0 and Scene 1 are animated *fireplace room* [18] and the other scenes are animated *Amazon Lumberyard Bistro* [19]. These scenes are rendered with the *Unity* game engine.

4.4.3 Pilot Study

We conduct a slider test and a random test in the pilot study. Each test consists of two steps:

1. the participant estimates the *Uniform Foveation Parameter* σ_{UF} which is acceptable for both the dominant eye and the non-dominant eye. We express this condition as $\sigma_d = \sigma_{nd} = \sigma_{UF}$;

2. the participant estimates the *Non-dominant Eye Foveation Parameter* σ_{NF} that results in the same overall visual perception as the uniform foveation, by

increasing the foveation level (reducing overall detail) of the rendering for the non-dominant eye. We express this condition as: $\sigma_d = \sigma_{UF}$, $\sigma_{nd} = \sigma_{NF}$.

Participants We recruited 17 participants (5 females) at least 18 years old with normal or corrected-to-normal vision via campus email lists and flyers. The majority of participants had some experience with virtual reality. None of the participants was involved with this project prior to the user study.

Slider Test The slider test allows the participants to navigate the foveation space by themselves. We conduct the test with five different scenes with one trial for each scene. We present the two-step study protocol as follows.

1. Estimation of σ_{UF}: In each trial, we first present the participant with the full-resolution rendering as a reference. Next, we present the participant with the same foveated rendering for both eyes and allow the participant to adjust the level of foveation by themselves: starting with the highest level of foveation, $\sigma_d = 3.0$, the participants progressively decrease the foveation level (with a step size of 0.2). The participants can switch between the foveated rendering result and the reference image back and forth until they arrive at the highest foveation level σ_{UF} (with the lowest overall level of detail) that is visually equivalent to the non-foveated reference.

2. Estimation of σ_{NF}: In each trial, we present the participant the foveated rendering with $\sigma_d = \sigma_{UF}$ for the dominant eye, and allow the participant to adjust the level of foveation for the non-dominant eye. Starting with $\sigma_{nd} = \sigma_{UF}$, the participant can progressively increase the foveation level (with a step size of 0.2)

until they reach the highest foveation level σ_{NF} that is perceptually equivalent to the foveated rendering with uniform foveation parameter σ_{UF}.

Random Test The random test allows the participant to score the quality of the foveated rendering with different parameters in a random sequence. We conduct the test with five different scenes with one trial for each scene. The two steps are detailed below.

1. Estimation of σ_{UF}: In each trial, we present the participant with two frames: (1) the full-resolution rendering, and (2) the foveated rendering with $\sigma_d = \sigma_{nd} = x$, where x is selected from the shuffled σ parameter array with σ ranging between 1.2 and 3.0 with a step size of 0.2. The two frames are presented in a random order. We ask the participants to score the difference between the two frames they observe with unlimited time to make their decision. The score S_{UF} contains five confidence levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual difference*, 3 represents *acceptable perceptual difference*, 2 represents *noticeable perceptual difference*, and 1 represents *significant perceptual difference*.

We use a pairwise comparison approach and the participants finish the trials with $1.2 \leq x \leq 3.0$ in a random order. We choose the maximum x which results in an evaluation of *perceptually identical* or *minimal perceptual difference* with respect to the full-resolution (non-foveated) rendering, *i.e.*,

$$\sigma_{UF} = \underset{x}{\text{argmax}} \ S_{UF}(x) \geq 4. \tag{4.5}$$

2. Estimation of σ_{NF}: In each trial, we present the participant with two frames: (1) foveated rendering with $\sigma_d = \sigma_{nd} = \sigma_{UF}$, and (2) foveated rendering with

$\sigma_d = \sigma_{UF}$, $\sigma_{nd} = x$, where x is selected from the shuffled parameter array with parameters ranging between σ_{UF} and 3.0 with a step size of 0.2. The two frames are presented in random order. We ask the participants to score the difference between the two frames with unlimited time to make their decisions. The score S_{NF} contains five confidence levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual imbalance*, 3 represents *acceptable perceptual imbalance*, 2 represents *noticeable perceptual imbalance*, and 1 represents *significant perceptual imbalance*.

We choose the maximum x that results in *perceptually identical* or *minimal perceptual imbalance* with respect to the uniform foveated rendering, *i.e.*,

$$\sigma_{NF} = \underset{x}{\text{argmax}} \ S_{NF}(x) \geq 4. \tag{4.6}$$

Results and Limitations of the Pilot Study From the pilot study, we find that: for most users, the dominant eye significantly dominates the visual perception and therefore eye-dominance-guided foveated rendering is likely to achieve significant speedup. From the one-way ANOVA test, we did not find a significant effect of the choice of scenes on the feedback (with $p = 0.8708 > 0.01$) for the slider test. Therefore, σ_{UF} and σ_{NF} do not correlate with the choice of scenes in a statistically significant manner. However, the pilot study yields a gap between the results of the slider test and the random test as shown in Figure 4.3.

We next present the potential reasons for this gap and our strategies for mitigating them:

1. Performing a *single* trial for each test per scene is likely to induce some inaccuracy in parameter estimation. To mitigate this for the main study, we

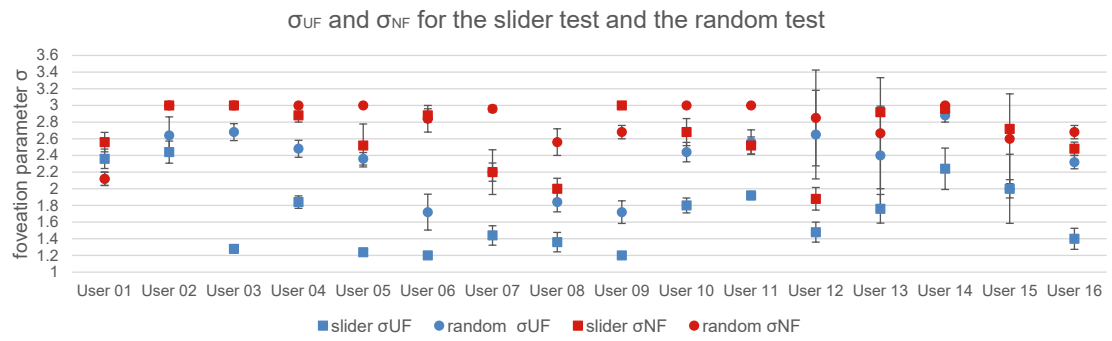


Figure 4.3: The average value of σ_{UF} and σ_{NF} in the slider and the random tests. The pilot study yields a gap between the results of the slider and the random tests.

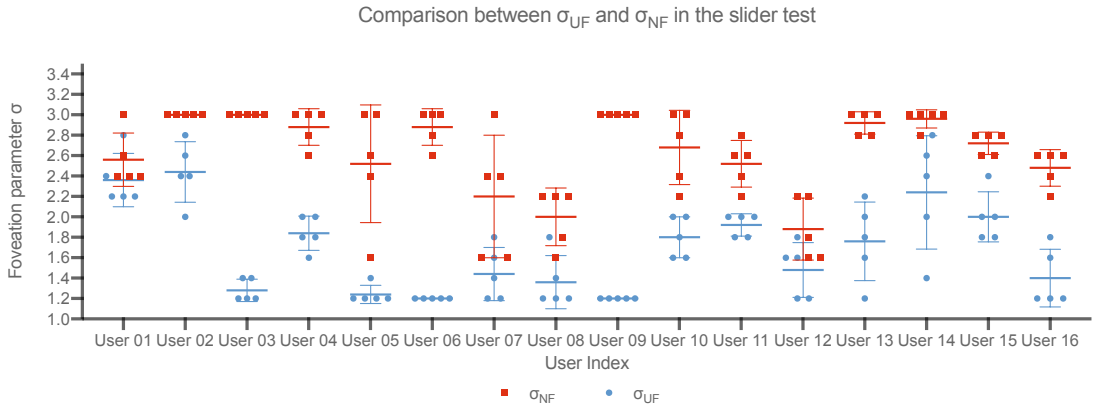


Figure 4.4: The result of the slider test of the pilot user study. We observe that σ_{NF} often reach our upper bound (3.0).

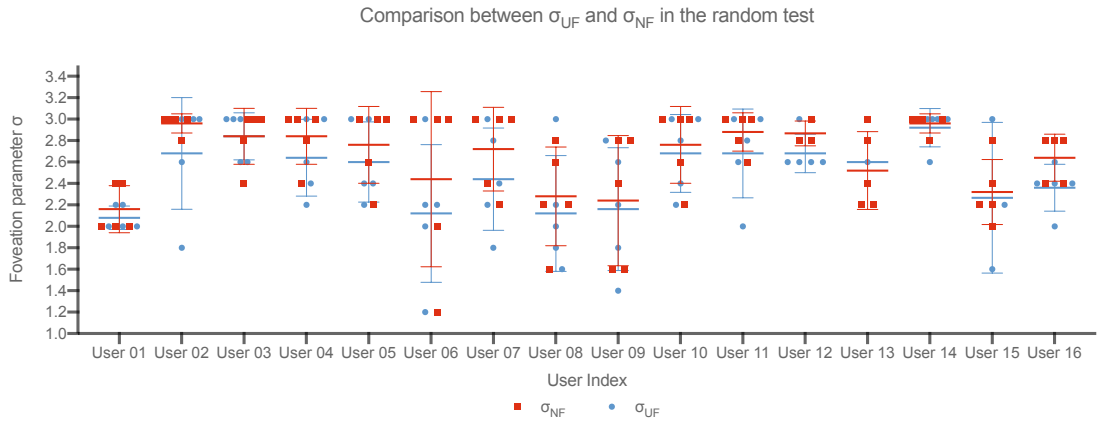


Figure 4.5: The result of the slider test of the random user study. We observe that σ_{NF} often reach our upper bound (3.0).

carry out *three* trials per scene per parameter;

2. In the pilot study, we use the maximum foveation parameter in Equations 4.5 and 4.6. We did this even if lower values of the foveation parameters led to an unacceptable score below 4. This was leading us to overestimate the foveation thresholds. In the main study, we use the greatest foveation parameter below which the user did not report an average score below 4. While this may reduce the speedups due to overall foveation, it will produce a higher perceptual quality;

3. We observe that σ_{NF} often reach our upper bound (3.0) – 42.5% in the slider test (as shown in Figure 4.4) and 60% in the random test (as shown in Figure 4.5). We have therefore increased the upper bound of σ_{NF} from 3.0 to 4.0 in the protocol of the main study;

4. In pilot study, we did not qualitatively evaluate the similarity in perceptual difference between EFR with the selected parameters and conventional foveated rendering (KFR) or regular rendering (RR). We therefore decide to add a quality evaluation in the main study.

Taking the above limitations and their mitigation strategies into account, we redesign the main study as described below.

4.4.4 Main Study

We conduct a slider test and a random test in the main study. There are three steps in both tests:

1. the participant estimates the *Uniform Foveation Parameter* σ_{UF};
2. the participant estimates the *Non-dominant Eye Foveation Parameter* σ_{NF};
3. the participant qualitatively evaluates whether the EFR frames with $\sigma_d = \sigma_{UF}$, $\sigma_{nd} = \sigma_{NF}$ are perceptually the same with RR or traditional (non-dominant) KFR.

We use Scene 3, Scene 5, and Scene 6 in Figure 4.2 for the parameter estimation in Steps 1 and 2 above. We use all the 10 scenes in Figure 4.2 for the quality evaluation.

Participants We recruited 11 participants (4 females) at least 18 years old with normal or corrected-to-normal vision via campus email lists and flyers. The majority of participants had some experience with virtual reality. None of the participants was involved with this project prior to the user study.

Slider Test The slider test allows the participant to navigate the foveation quality space by themselves.

1. Estimation of σ_{UF}: We conduct the test on three scenes with three trials per scene. Therefore, there are 9 tests in total. For the n-th trial of scene m, we first present the participant with the full-resolution rendering, as a reference. Next, we present the participant with the same foveated rendering for both eyes and allow the participant to adjust the level of foveation by themselves: starting with the highest level of foveation, $\sigma_d = 3.0$, the participants progressively decrease the foveation level (with a step size of 0.2). The participant can switch between the foveated rendering result and the reference image back and forth until they can identify the lowest foveation level $\sigma_{UF}(m, n)$ that is visually equivalent to the non-foveated reference. We calculate the mean uniform foveation parameter for scene m:

$$\sigma_{UF}(m) = \frac{1}{3} \sum_{n=1,2,3} \sigma_{UF}(m, n). \tag{4.7}$$

We calculate the overall mean uniform foveation parameter:

$$\sigma_{UF} = \frac{1}{3} \sum_{m=1,2,3} \sigma_{UF}(m). \tag{4.8}$$

2. Estimation of σ_{NF}: We conduct the test on three scenes with three trials per scene. Therefore, there are 9 tests in total. For the n-th trial of scene m, we present the participant the foveated rendering with $\sigma_d = \sigma_{UF}(m)$ for the dominant eye, and allow the participant to adjust the level of foveation for the non-dominant eye: starting with $\sigma_{nd} = \sigma_{UF}(m)$, the participants can progressively increase the foveation level (with a step size of 0.2) until they reach the highest foveation level

$\sigma_{NF}(m, n)$ that is perceptually equivalent to the foveated rendering with uniform foveation parameter $\sigma_{UF}(m)$.

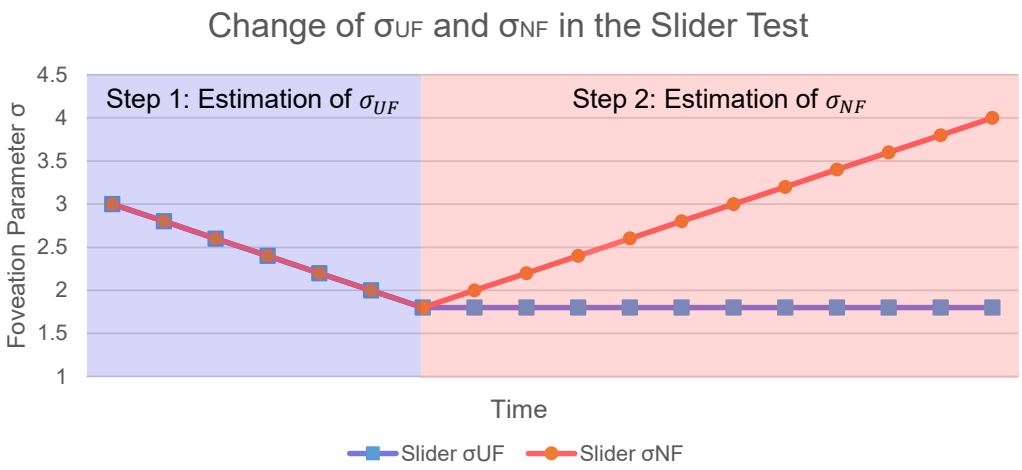


Figure 4.6: Change of parameter σ_d and σ_{nd} in the slider test. In Step 1, estimation of σ_{UF}, we present the participant with the same foveated rendering for both eyes and the participant progressively decrease the foveation level until $\sigma_d = \sigma_{UF}(m)$. In Step 2, estimation of σ_{NF}, we present the participant the foveated rendering with $\sigma_d = \sigma_{UF}(m)$ for the dominant eye, and allow the participant to adjust the level of foveation for the non-dominant eye. The participant can progressively increase the foveation level until they reach the highest foveation level.

Figure 4.6 shows the change of parameters σ_{UF} and σ_{NF} in Step 1: Estimation of σ_{UF} and Step 2: Estimation of σ_{NF} in each trial. We calculate the mean uniform foveation parameter for scene m:

$$\sigma_{NF}(m) = \frac{1}{3} \sum_{n=1,2,3} \sigma_{NF}(m, n). \tag{4.9}$$

We calculate the overall mean uniform foveation parameter:

$$\sigma_{NF} = \frac{1}{3} \sum_{m=1,2,3} \sigma_{NF}(m). \tag{4.10}$$

3. Quality evaluation: We conduct the A/B test on 10 scenes with two comparisons (EFR vs. KFR and EFR vs. RR) per scene, and 1 trial per scene per comparison. There are 20 trials in total. For scene m, we present the participant with two frames: (1) EFR with $\sigma_d = \sigma_{UF}$, $\sigma_{nd} = \sigma_{NF}$ and (2) RR or KFR with $\sigma_d = \sigma_{nd} = \sigma_{UF}$. The two frames are presented in random order. Then we ask the participants to score the difference between the two frames they observed with unlimited time to make their decisions. The score $S(m)$ contains five confidence levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual difference*, 3 represents *acceptable perceptual difference*, 2 represents *noticeable perceptual difference*, and 1 represents *significant perceptual difference*.

Random Test The random test allows the participant to score the quality of foveated rendering with different parameters in a random sequence.

1. Estimation of σ_{UF}: We conduct the test on three scenes with 10 parameters per scene, each with three trials. Therefore, there are 90 tests in total. For the n-th trial of scene m, we present the participant with two frames: (1) the full-resolution rendering, and (2) the foveated rendering with $\sigma_d = \sigma_{nd} = x$, where x is selected from the shuffled parameter array with parameters ranging between 1.2 and 3.0 with a step size of 0.2. The two frames are presented in a random order. Then, we ask the participant to score the difference between the two frames they observed with unlimited time to make their decision. The score $S_{UF}(m, n, x)$ contains five confidence

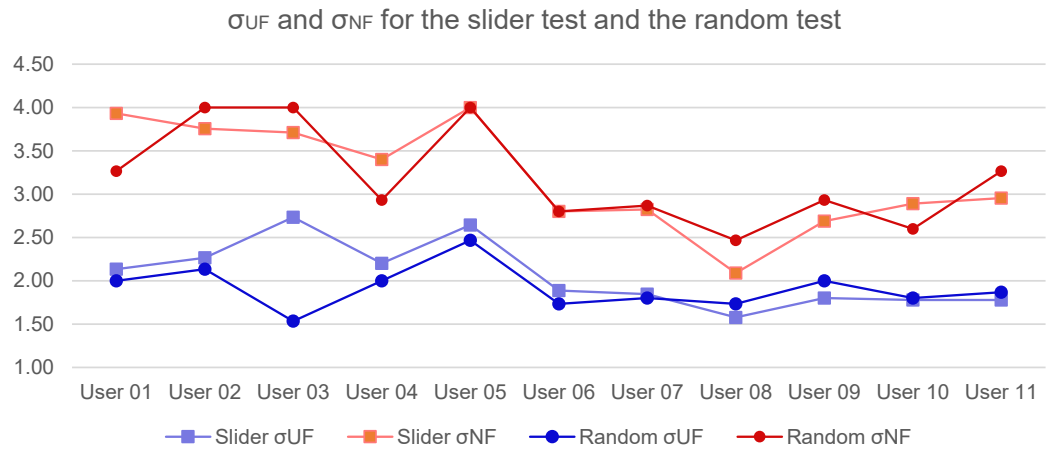


Figure 4.7: The average value of σ_{UF} and σ_{NF} in the slider test and the random test. A paired T-test reveals no significant difference ($p = 0.8995 > 0.01$) between the result of the slider test and the result of the random test.

levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual difference*, 3 represents *acceptable perceptual difference*, 2 represents *noticeable perceptual difference* and 1 represents *significant perceptual difference*.

When the process is finished, we calculate the average score of all the trials for scene m with foveation parameter x:

$$S_{UF}(m, x) = \frac{1}{3} \sum_{n=1,2,3} S_{UF}(m, n, x). \tag{4.11}$$

We choose the minimum x which results in an evaluation of *perceptually identical* or *minimal perceptual difference* with respect to the full-resolution (non-foveated) rendering as $\sigma_{UF}(m)$, i.e.,

$$\sigma_{UF}(m) = \underset{x}{\operatorname{argmin}} \ S_{UF}(m, x) \geq 4. \tag{4.12}$$

We calculate σ_{UF} using Equation 4.8.

2. Estimation of σ_{NF}: We conduct the test on three scenes with Q parameters for scene m and three trials per scene per parameter. We compute Q using Equation 4.13 with $\sigma_{max} = 4.0$ in the main study.

$$Q = \frac{\sigma_{max} - \sigma_{UF}(m)}{0.2} + 1 \tag{4.13}$$

For the n-th trial of scene m, we present the participant with two frames: (1) foveated rendering with $\sigma_d = \sigma_{nd} = \sigma_{UF}(m)$, and (2) foveated rendering with $\sigma_d = \sigma_{UF}(m)$, $\sigma_{nd} = x$, where x is selected from the shuffled parameter array with Q parameters ranging between $\sigma_{UF}(m)$ and $\sigma_{max} = 4.0$ with a step size of 0.2. The two frames are presented in a random order. Then we ask the participants to score the difference between the two frames they observed with unlimited time to make their decisions. The score $S_{NF}(m, n, x)$ contains five confidence levels: 5 represents *perceptually identical*, 4 represents *minimal perceptual imbalance*, 3 represents *acceptable perceptual imbalance*, 2 represents *noticeable perceptual imbalance* and 1 represents *significant perceptual imbalance*.

When the process is finished, we calculate the average score of all the trials for scene m with foveation parameter x:

$$S_{NF}(m, x) = \frac{1}{3} \sum_{n=1,2,3} S_{NF}(m, n, x). \tag{4.14}$$

We choose the minimum x which results in an evaluation of *perceptually identical* or *minimal perceptual imbalance* with respect to the full-resolution (non-foveated)

rendering as $\sigma_{NF}(m)$, i.e.,

$$\sigma_{NF}(m) = \underset{x}{\text{argmin}} \ S_{NF}(m, x) \geq 4. \tag{4.15}$$

We calculate σ_{NF} using Equation 4.10.

3. Quality evaluation: The quality evaluation is the same as that of the slider test.

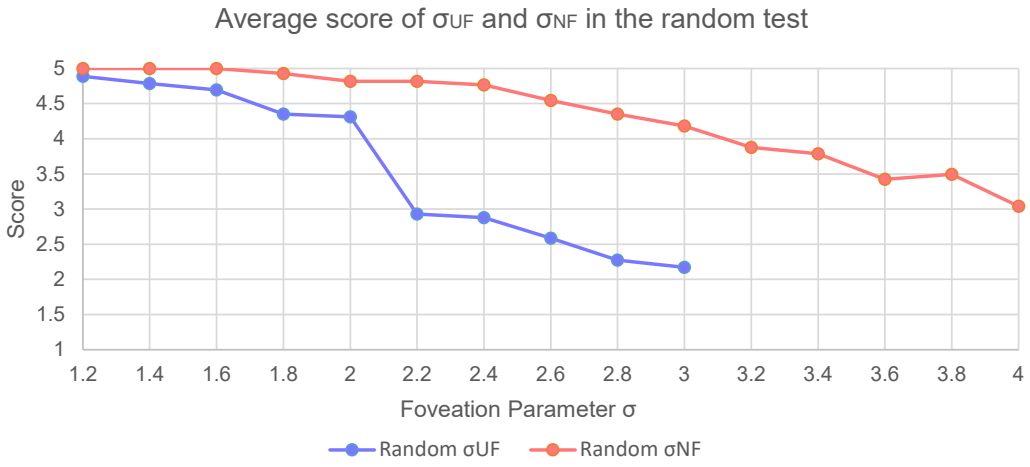


Figure 4.8: The average score in Step 1 (Estimation of σ_{UF}) and Step 2 (Estimation of σ_{NF}) over different scenes and different users in the random test. To achieve *perceptually identical* and *minimal perceptual difference* between regular rendering and foveated rendering, we therefore choose $\sigma_{UF} = 2.0$ and $\sigma_{NF} = 3.0$ as our desired parameters.

4.4.5 Validity Test

Eye Tracking Data Analysis We collected eye-tracking data from the FOVE HMD and would like to use it as a high-level validation to ensure that the participants are focusing at the desired fovea location.

However, we have noticed obvious tracking errors during the process: sometimes the eye-tracker fails to capture the movement of gaze and sometimes the tracked gaze position changes when the user blinks while focusing at the center of the screen. We also need to ensure that the users are paying attention to the user study instead of randomly choosing the answers. Therefore, it may not be ideal to solely depend on eye tracking results for judging the participants' focus. We also use the participant's performance with respect to the ground truth data to determine the accuracy and participant focus. We discuss this next.

Controlling for Lack of Attention and Exhaustion We randomly inserted 30% of the trials to be validation trials in the random test to ensure the validity of the data in the pilot study and the main study. For uniform foveation parameter estimation, we presented the participants with identical full-resolution rendering results for both comparison frames as validation trials; for non-dominant eye foveation parameter estimation, we presented the participants with identical rendering results with $\sigma_d = \sigma_{nd} = \sigma_{UF}$ for both comparison frames as validation trials. If the participant declared these validation trials to have a low score for similarity (3 or lower), we would ask the participant to pause and take a break for at least 30 seconds,

and then continue the user study. Meanwhile, we would record this choice as an *error*. If $error \geq 5$ in the random test, we would terminate the user study and discard the data of this participant. Based on this protocol, we discard one participant from the pilot study and mark the remaining 16 participants as valid data. All the 11 participants in the main study passed the validation trials.

Comparison	Score = 1	Score = 2	Score = 3	Score = 4	Score = 5
Slider: EFR vs. RR	0.00%	2.73%	8.18%	17.27%	71.82%
Slider: EFR vs. KFR	0.00%	4.55%	10.91%	30.00%	54.55%
Random: EFR vs. RR	0.00%	0.00%	0.91%	14.55%	84.55%
Random: EFR vs. KFR	0.00%	0.91%	3.64%	25.45%	70.00%

Table 4.1: The score frequency for different comparisons in the slider test and the random test. We notice that $P(score \geq 4) \geq 85\%$ for both comparisons in the slider test and that $P(score \geq 4) \geq 95\%$ for both comparisons in the random test. The result indicates the generalizability of eye-dominance-guided foveated rendering.

4.5 Results and Acceleration

In our main user study, the number of errors in the attention and exhaustion checking is less than 5 over all the participants. We use the results from all the 11 participants for data analysis.

4.5.1 Parameters Estimated with Different Scenes

We conducted a one-way ANOVA test [98, 99] of the null hypothesis that the choice of scenes has no effect on the feedback of the participants. With the slider test, we did not find a significant effect of the choice of scenes on the feedback (with $p = 0.9782 > 0.05$).

4.5.2 Results of σ_{UF} and σ_{NF}

For user i, we consider the averages of σ_{UF} (Equation 4.8) and σ_{NF} (Equation 4.10) over different scenes as the per-user foveation parameter for the dominant eye $\sigma_{UF,i}$ and non-dominant eye $\sigma_{NF,i}$. We present these results in Figure 4.7.

We first verified if there is a significant difference of the measured parameters (σ_{UF} and σ_{NF}) between the slider test and the random test. With a paired T-test, we did not find a significant effect between the slider test and the random test (with $p = 0.8995 > 0.05$).

The paired T-test shows that the EFR parameters are stable with different experimental setups. We therefore take the average of slider test and the random test as the final parameters to test the rendering acceleration.

We further conducted statistical analysis on the difference between σ_{UF} and σ_{NF}. With a paired T-test, we found a significant effect that the foveation parameter σ_{UF} required for the non-dominant eye is higher than the foveation parameter σ_{NF} for the dominant eye (with $p = 7.0530 \times 10^{-10} < 0.05$). Hence, we reach a conclusion that the disparity between the visual acuity in the dominant eye and the non-dominant

eye is significantly different for the users.

For the random test, we also present the average score in Step 1 (estimation of σ_{UF}) and Step 2 (estimation of σ_{NF}) over different scenes and different users as shown in Figure 4.8. We notice that both S_{UF} and S_{NF} decrease with the increase of the foveation parameter. To achieve *perceptually identical* and *minimal perceptual difference* between regular rendering and foveated rendering for most users, we may choose $\sigma_{UF} = 2.0$ and $\sigma_{NF} = 3.0$ as the desired parameters.

4.5.3 Quality Evaluation

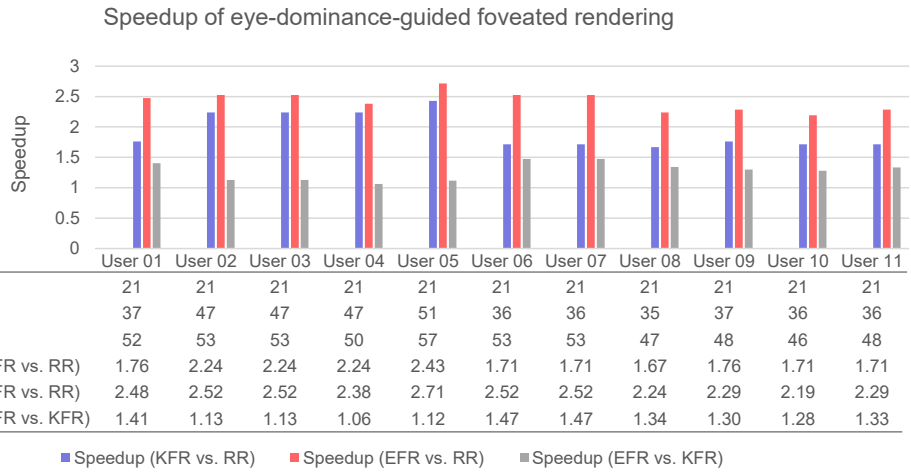
We analyzed whether there exists a significant difference of the quality evaluation results between the slider test and the random test. With the paired T-test, we did not find a significant effect between the slider test and the random test (with $p = 0.8629 > 0.05$).

We further verified if there exists a significant difference of the quality evaluation results between the experiment of EFR vs. KFR and the experiment of EFR vs RR. With a paired T-test, we found no significant difference between the result of the two experiments (with $p = 0.9410 > 0.05$).

The frequency from $score = 1$ to $score = 5$ is shown in Table 4.1. We notice that $P(score \geq 4) \geq 85\%$ for both comparisons in the slider test and that $P(score \geq 4) \geq 95\%$ for both comparisons in the random test. The result indicates the generalizability of eye-dominance-guided rendering. We can get acceptable perceptual quality that on different scenes with the measured parameters from the

user study.

4.5.4 Rendering Acceleration



	User 01	User 02	User 03	User 04	User 05	User 06	User 07	User 08	User 09	User 10	User 11
RR (fps)	21	21	21	21	21	21	21	21	21	21	21
KFR (fps)	37	47	47	47	51	36	36	35	37	36	36
EFR (fps)	52	53	53	50	57	53	53	47	48	46	48
Speedup (KFR vs. RR)	1.76	2.24	2.24	2.24	2.43	1.71	1.71	1.67	1.76	1.71	1.71
Speedup (EFR vs. RR)	2.48	2.52	2.52	2.38	2.71	2.52	2.52	2.24	2.29	2.19	2.29
Speedup (EFR vs. KFR)	1.41	1.13	1.13	1.06	1.12	1.47	1.47	1.34	1.30	1.28	1.33

Figure 4.9: The measured frame-rates (in fps) and the speedups. The speedups of the eye-dominance-guided foveated rendering (EFR) compared with the original kernel foveated rendering (KFR) ranges between 1.06× and 1.47× with an average speedup of 1.35×. The speedups of EFR compared with regular rendering (RR) ranges between 2.19× and 2.71× with an average speedup of 2.38×.

We have implemented the eye-dominance-guided foveated rendering pipeline in $C++$ 11 and OpenGL 4 on NVIDIA GTX 1080 to measure the rendering acceleration. We report our speedups based on the sophisticated *Amazon Lumberyard Bistro* dataset [19] at the resolution of 1280×1440 per eye. The frame-rates and speedups of the original kernel foveated rendering (KFR) and the eye-dominance-guided foveated rendering (EFR) compared with traditional regular rendering (RR) are

shown in Figure 4.9. The speedup of the eye-dominance-guided foveated rendering compared with the original kernel foveated rendering ranges between 1.06× and 1.47× with an average speedup of 1.35×.

Chapter 5: Hand Mesh Reconstruction from RGB Images

5.1 Overview

Hands play an important role in our daily life. An approach that could detect the shape and gesture of the human hand from RGB images would enable new applications in virtual and augmented reality [100, 101] and human-computer interaction [102–105]. However, the current state-of-the-art approaches do not provide a good solution because of the depth and scale ambiguities. In recent years, deep learning is playing an important role in visual interactions [106] by providing success in hand pose reconstruction from a depth image [107, 108], hand pose reconstruction from a RGB image [109, 110], as well as hand mesh reconstruction from a RGB image [20, 21, 111]. Here, I focus on the hand reconstruction with only RGB images as input. This is significantly more challenging than hand reconstruction when the depth data is also available as an auxiliary feature.

Our contributions include:

- building a new dataset of the hand by fitting a hand model to 74,715 3D joint annotations in the Panoptic Studio dataset to solve the problem of sparse training annotation;

- designing an end-to-end neural network, which accepts a RGB image and the auxiliary features as inputs and predicts a 3D hand mesh of the right hand that could be projected onto the RGB image and match the 2D hand in shape and pose;
- evaluating our research in terms of 3D pose estimation on various public datasets.

5.2 Related Work

5.2.1 Hand Model

Taylor *et al.* [112] presents a method for acquiring dense shape and deformation from a single monocular depth sensor. Khamis *et al.* [113] propose the first learning-based subject-specific hand shape variation from scans with linear blend skinning. The MANO (hand Model with Articulated and Non-rigid defOrmations) model [114] is a hand deformation model based on the SMPL [115] model for human bodies. MANO models both hand shape and pose, thus generating realistic posed meshes.

5.2.2 Hand Skeleton Reconstruction from Multiview

The multi-view image processing techniques could be utilized to refine the hand reconstruction models. Campos and Murray [116] use the relevance vector machine-based learning for hand pose recovery. Sridhar *et al.* [117] propose a real-time markerless hand tracking approach which employs an implicit hand shape

representation based on sum of anisotropic Gaussians and minimizes the sum of the pose fitting energy. Simon *et al.* [109] propose a multi-view hand pose prediction system which uses a pretrained weak predictor to estimate the hand pose and retrain an improved detector by annotating the failed detection with the re-projected successful detection.

5.2.3 Hand Mesh Reconstruction from Singleview

Boukhayma *et al.* [21] predict both 3D hand shape and pose from RGB images with heatmaps in the wild. Their pipeline consists of the concatenation of a deep convolutional encoder (ResNet-34), and a MANO decoder. Zhang *et al.* [118] concatenate the autoencoder with an iterative regression block and refine the estimated parameter iteratively. Ge *et al.* [111] and Baek *et al.* [119] use a stacked hourglass network to predict heatmaps as the auxiliary feature and predict the hand mesh with a convolutional neural network. Kulon *et al.* [20] learn the prior on 3D hand shapes by training an autoencoder with intrinsic graph convolutions performed in the spectral domain. Kulon *et al.* [120] solve the problem of sparse supervision by gathering a large-scale dataset of hand action in YouTube videos and use it as a source of weak supervision. They propose an autoencoder system with ResNet-50 as the encoder and a spatial mesh convolutional decoder.

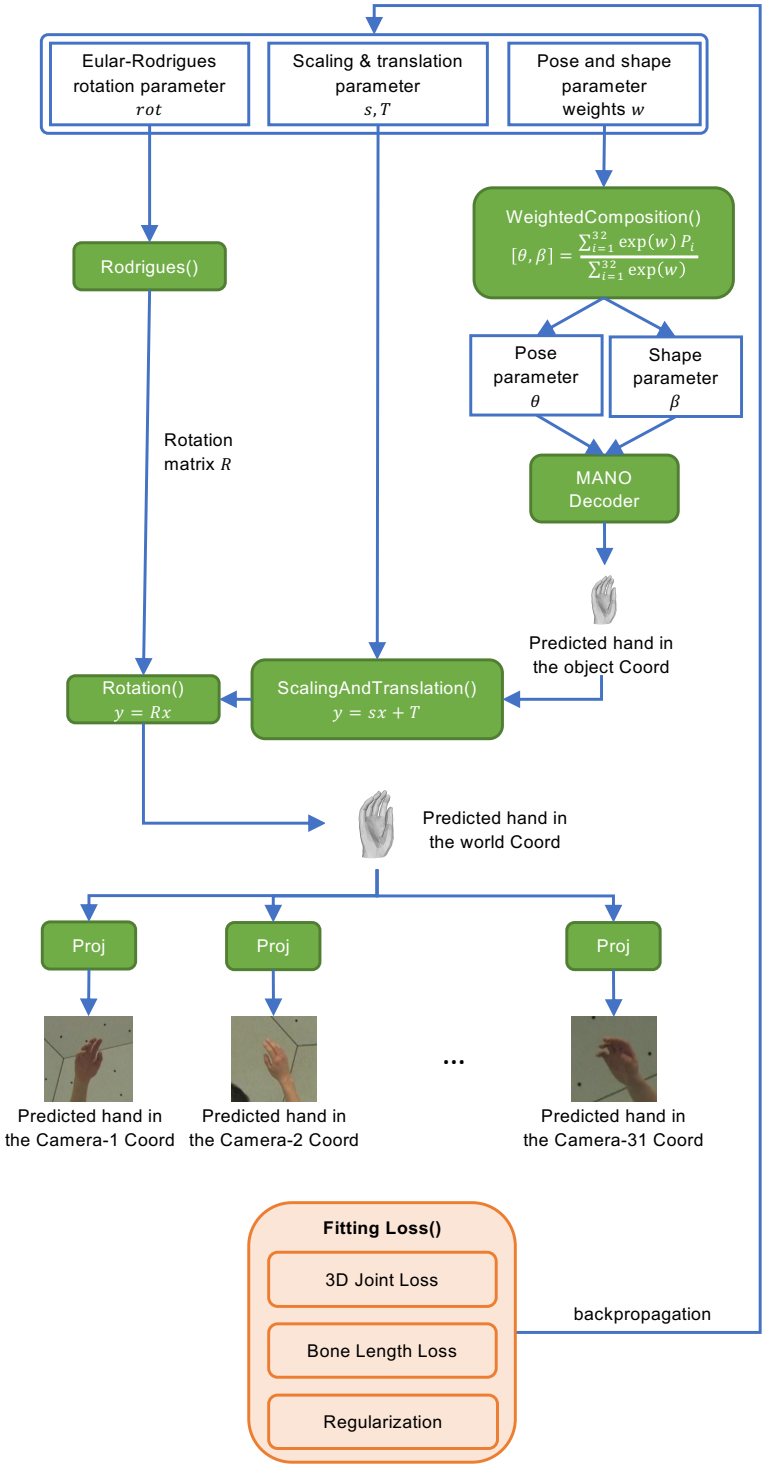


Figure 5.1: The pipeline of the ground truth mesh generation.

5.3 Estimation of the Hand Mesh

5.3.1 Datasets

We use the CMU Panoptic Dataset [121] for hand mesh estimation. For each scene, there are 31 videos captured with HD synchronized cameras. The dataset also contains camera information, visibility of the hands from the target cameras, and the 3D joint annotations in the world space.

5.3.2 Pipeline

The pipeline of the estimation of the hand mesh is shown in Figure 5.1. For each hand image in the Panoptic Dataset with 3D joint annotation J_I, we predict a 3D hand mesh generated with the MANO model by minimizing the error between the 3D joint annotation J_I in the image coordinates and the predicted 3D joints in the image coordinates $\hat{J}_I$ by predicting the parameters for rotation rot, shape β, pose θ, scaling s, and translation t. By accepting β and θ as inputs, the MANO decoder $\mathcal{M}(rot, \theta, \beta)$ decodes a hand mesh $\hat{H}_{\text{MANO}}$ in the local object coordinates (including the hand joint information $\hat{J}_{\text{MANO}}$ and the hand vertex information $\hat{V}_{\text{MANO}}$ in the local object coordinates), $\hat{H}_{\text{MANO}}$ is projected to the world coordinates $\hat{H}_W$ (including hand joints $\hat{J}_W$ and hand vertices $\hat{V}_W$ in the local object coordinates) with the guidance of r, s and T. And $\hat{H}_W$ is projected to the image coordinates $\hat{H}_I$ (including hand joints $\hat{J}_I$ and hand vertices $\hat{V}_I$ in the local object coordinates) to match the hand in the RGB image with ground truth camera extrinsic parameters

R and T and camera intrinsic parameter K.

$$\begin{aligned}
&\{rot^*, \theta^*, \beta^*, s^*, t^*\} \\
&= \underset{rot,\theta,\beta,s,t}{\operatorname{argmin}} L(J_I, \hat{J}_I) \\
&= \underset{rot,\theta,\beta,s,t}{\operatorname{argmin}} L(J_I, \mathcal{P}(\hat{J}_{\text{MANO}}, \hat{s}, \hat{t}, K, R, T)) \\
&= \underset{rot,\theta,\beta,s,t}{\operatorname{argmin}} L(J_I, \mathcal{P}(\mathcal{M}(rot, \hat{\theta}, \hat{\beta}), \hat{s}, \hat{t}, K, R, T)),
\end{aligned} \tag{5.1}$$

Instead of modelling the joint angles as free variables, which can lead to physically implausible hand, we constrain the pose parameters and the shape parameters to lie in the convex hull of the pre-computed cluster centers [120]. We obtained the cluster centers by applying k-means on the FreiHAND [122] dataset and cluster the pose parameters and the shape parameters into 32 clusters. In the fitting process, we predict the angle weights $w_{\text{ClusterID}}$ with ClusterID $\in [1, 32]$ and calculate the pose and shape parameters as shown in Equation 5.2.

$$[\theta, \beta] = P(w) = \frac{\sum_{\text{ClusterID}=1}^{32} exp(w_{\text{ClusterID}}) P_{\text{ClusterID}}}{\sum_{\text{ClusterID}=1}^{32} exp(w_{\text{ClusterID}})} \tag{5.2}$$

where $F(X_{\text{MANO}}, s, t, K, R, T)$ finishes the scaling and translation of X_{MANO}, which could be the predicted joints $\hat{J}_{\text{MANO}}$ or the predicted vertices $\hat{V}_{\text{MANO}}$:

$$\begin{aligned}
\mathcal{P}(X_{\text{MANO}}, s, t, K, R, T) =& K[R|T] \cdot X_W \\
=& K[R|T] \cdot (sX_{\text{MANO}} + t)
\end{aligned} \tag{5.3}$$

5.3.3 Training Objective

We combine multiple losses to predict the parameters: a 3D joint loss in the camera coordinates L_{joint}, a bone length loss L_{bone}, and a regularization term L_{reg}.

$$L = \alpha_{joint} L_{joint} + \alpha_{bone} L_{bone} + \alpha_{reg} L_{reg} \tag{5.4}$$

3D joint loss L_{joint}: We project the predicted hand joints from the world coordinates $\hat{J}_W$ to the i-th camera coordinates $\hat{J}_{C,i}$, calculate the joint error between $\hat{J}_{C,i}$ and the ground truth hand joints in the camera coordinates $J_{C,i}$ by taking the weighted sum of the joint error of the hand root (ROOT), metacarpophalangeal (MCP), proximal interphalangeal (PIP), distal interphalangeal (DIP) and the finger tips (TIP).

Finally we calculate the sum of the joint error of all the 31 camera coordinates as shown in Equation 5.5.

$$\begin{aligned} L_{joint} &= \sum_{k=1}^{31} L(J_{C,k} - \hat{J}_{C,k}) \\ &= \sum_{k=1}^{31} (\alpha_{\text{MCP}} ||J_{\text{MCP}} - \hat{J}_{\text{MCP}}||^2 \\ &\quad \alpha_{\text{PIP}} ||J_{\text{PIP}} - \hat{J}_{\text{PIP}}||^2 + \\ &\quad \alpha_{\text{DIP}} ||J_{\text{DIP}} - \hat{J}_{\text{DIP}}||^2 + \\ &\quad \alpha_{\text{TIP}} ||J_{\text{TIP}} - \hat{J}_{\text{TIP}}||^2 + \\ &\quad \alpha_{\text{ROOT}} ||J_{\text{ROOT}} - \hat{J}_{\text{ROOT}}||^2) \end{aligned} \tag{5.5}$$

Bone length loss L_{bone}: The 21 hand joints could be interpreted as 20 groups of bone edges (BE). The loss of bone length ensures that the lengths of the bones

coincide with the ground truth hand bone lengths.

$$L_{bone} = \sum_{k=1}^{31} \sum_{(i,j)\in BE} |J_{C,k,i} - \hat{J}_{C,k,j}| \tag{5.6}$$

Regularization L_{reg}: The regularization term ensures that the predicted hand is physically plausible.

$$L_{reg} = \alpha_\theta ||\theta||^2 + \alpha_\beta ||\beta||^2 \tag{5.7}$$

The regularization parameters $\alpha_\theta = 10^{-1}$ and $\alpha_\beta = 10^3$ are chosen experimentally.

5.3.4 Optimization

We use the Adam optimizer [123] with different learning rates for the rotation rot, scaling s, translation t, and the MANO parameter weights $W_{\#Cluster\times(|\beta|+|\theta|)}$ with $\#Cluster = 32$. The optimizer is used with small learning rate decay (multiplicative factor of 0.95) when $loss \leq 150$. We fit 2048 groups of frames per batch on GeForce RTX 2080 which takes on average 40 min.

5.3.5 Evaluation Metric and Result

We use F-score [124] to evaluate the quality of the generated hand mesh and we accept $F@10mm = 1$ as acceptable.

We used 11 scenes from the Panoptic Dataset containing $241,008$ valid frames with physically plausible hands and 3D annotation in 31 cameras for the fitting and got $48,955$ mesh with $F@10mm = 1$ in the multiview fitting and $47,954$ mesh with

Figure 5.2: The qualitative results of hand mesh estimation from joints.

$F@10mm = 1$ in the temporal-multiview fitting. The qualitative results are shown in Figure 5.2.

5.4 Hand Reconstruction from RGB Images

5.4.1 Overview

We propose to reconstruct a 3D hand mesh from a single RGB image as illustrated in Figure 5.3. The RGB image of the hand $\mathcal{I}$ is passed to a pre-trained multi-stage convolutional neural network [125] to predict the heatmaps $\mathcal{H}$ for the 21 hand joints. The RGB image $\mathcal{I}$ and the heatmaps $\mathcal{H}$ are stacked and encoded to camera embedding and mesh embedding in the *Resnet-34* [126] encoder. The mesh embedding is decoded into a hand mesh $\tilde{H}$ in the camera coordinates, and projected to the image coordinates using a weak perspective camera created with the camera embedding.

5.4.2 Encoder

The encoder takes an RGB image $\mathcal{I}$ and the heatmaps $\mathcal{H}$ as the input and uses the *Resnet-34* [126] network pretrained on the *ImageNet* [127] classification task. The output of the encoder is a vector $rot, s, T, \theta, \beta \in \mathbb{R}^{61}$. The mesh embedding containing the pose parameter θ and the shape parameter β are passed to the decoder. And the camera embedding including the Rodrigues rot, scaling s, and translation T is fed into a weak perspective camera system, which projects the hand mesh from the camera coordinates to the image coordinates.

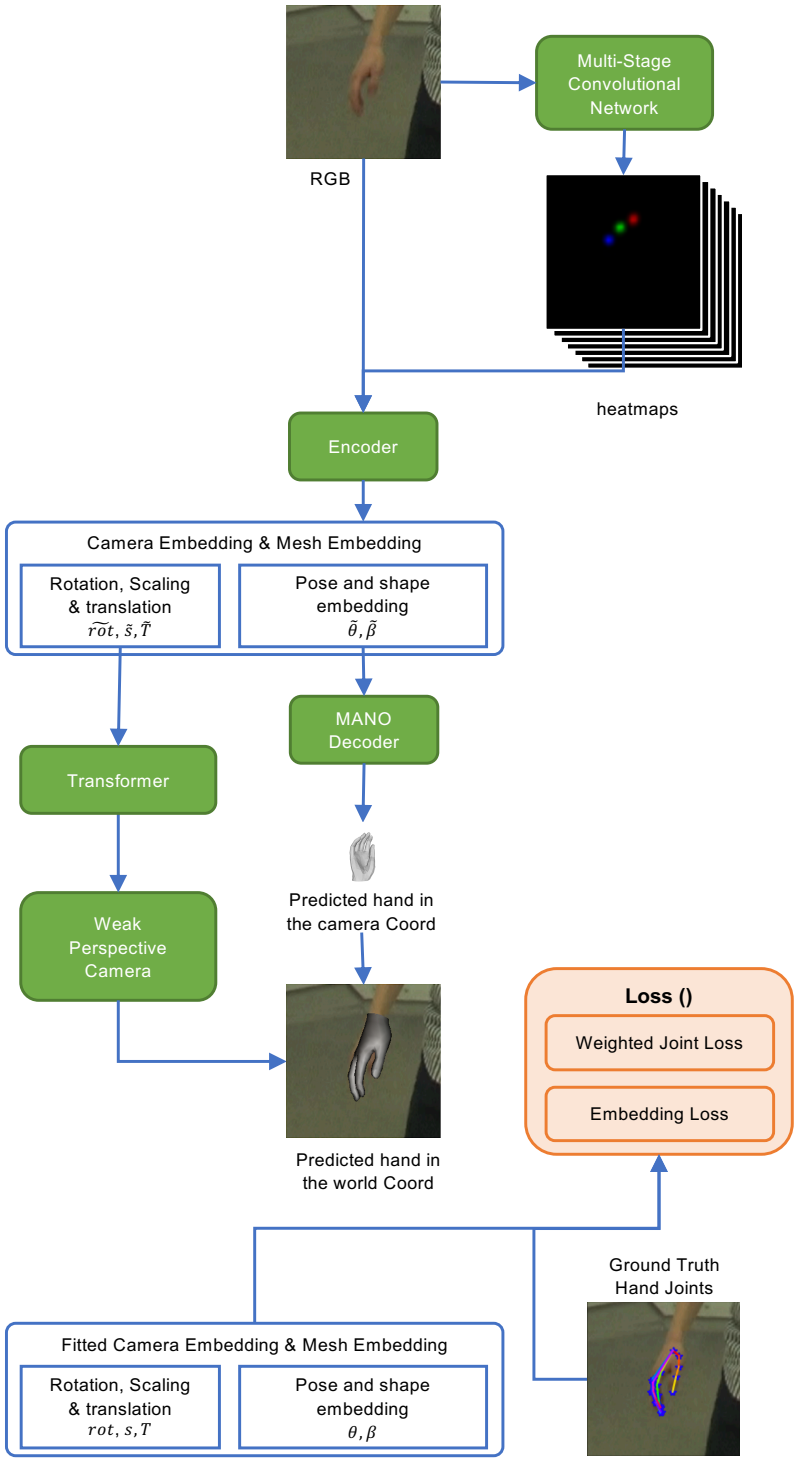


Figure 5.3: The pipeline of the hand Reconstruction from RGB Images.

5.4.3 Decoder

The differentiable MANO decoder is used to recover 3D hand meshes from the mesh embedding. The output is a MANO hand deformation in the camera coordinates.

5.4.4 Training Objective

We train the network with the estimated dense supervision and we combine multiple losses:

2D joint loss L_{joint}: We calculate the 2D joint loss in the image coordinates, which is the joint error between the predicted 2D joints and the ground truth 2D joints:

$$L_J = ||J_{2D} - \tilde{J_{2D}}||_2 \tag{5.8}$$

Embedding loss $L_{embedding}$: The embedding loss $L_{embedding}$ enforces the consistency between the predicted embedding $\{\tilde{rot}, \tilde{s}, \tilde{T}, \tilde{\theta}, \tilde{\beta}\}$ and the ground truth embedding $\{rot, s, T, \theta, \beta\}$ estimated from the fitting process as described in Chapter 5.3:

$$\begin{aligned} L_{embedding} = \alpha_s||s - \tilde{s}||_2^2 + \alpha_T||T - \tilde{T}||_2^2 + \alpha_{rot}||rot - \tilde{rot}||_2^2 \\ + \alpha_\theta||\theta - \tilde{\theta}||_2^2 + \alpha_\beta||\beta - \tilde{\beta}||_2^2, \end{aligned} \tag{5.9}$$

where $\alpha_s = 10^{-2}$, $\alpha_T = 10^{-4}$, $\alpha_{rot} = 10^2$, $\alpha_\beta = \alpha_\theta = 1$.

5.5 Results and Comparisons

5.5.1 Experimental Setup

Implementation: Our network is implemented in PyTorch [128]. Before training, the weights of the encoder are initialized with the weights of an image classification model pre-trained on the *ImageNet* [127] dataset. The network is trained end-to-end using the Adam optimizer [123] with mini-batches of size 32. The learning rate is set as 10^{-5} and we keep the default values for other parameters.

Training and Testing: For the Panoptic dataset with fitted ground truth, we hold out 6284 randomly-selected samples as the test data, 6393 samples as the validation data, and use the remaining 62038 samples as the training data. We use RGB image crops of human hands with a resolution of 256×256 as the input. The ground truth hand joints are used to find the tightest bounding box $B_0(w_0, h_0)$, and the images are cropped with a squared patch of size $B_1(2.2max(w_0, h_0), 2.2max(w_0, h_0))$ centered at the same 2D position as B_0. The new patches resized to 256×256 are used as the input RGB images. We report the mesh estimation result using the autoencoder trained for 450 epochs.

5.5.2 Quantitative Evaluation of 3D Hand Mesh Estimation

We report the performance of the 3D hand mesh estimation with two metrics:

- 2D/3D PCK: the percentage of correct keypoints (PCK) of which the Euclidean

Figure 5.4: Qualitative evaluation results of the fitted hand mesh (the second column), our hand reconstruction approach (the third column), Kulon *et al.* [20] (the fourth column), and Boukhayma *et al.* [21] (the fifth column).

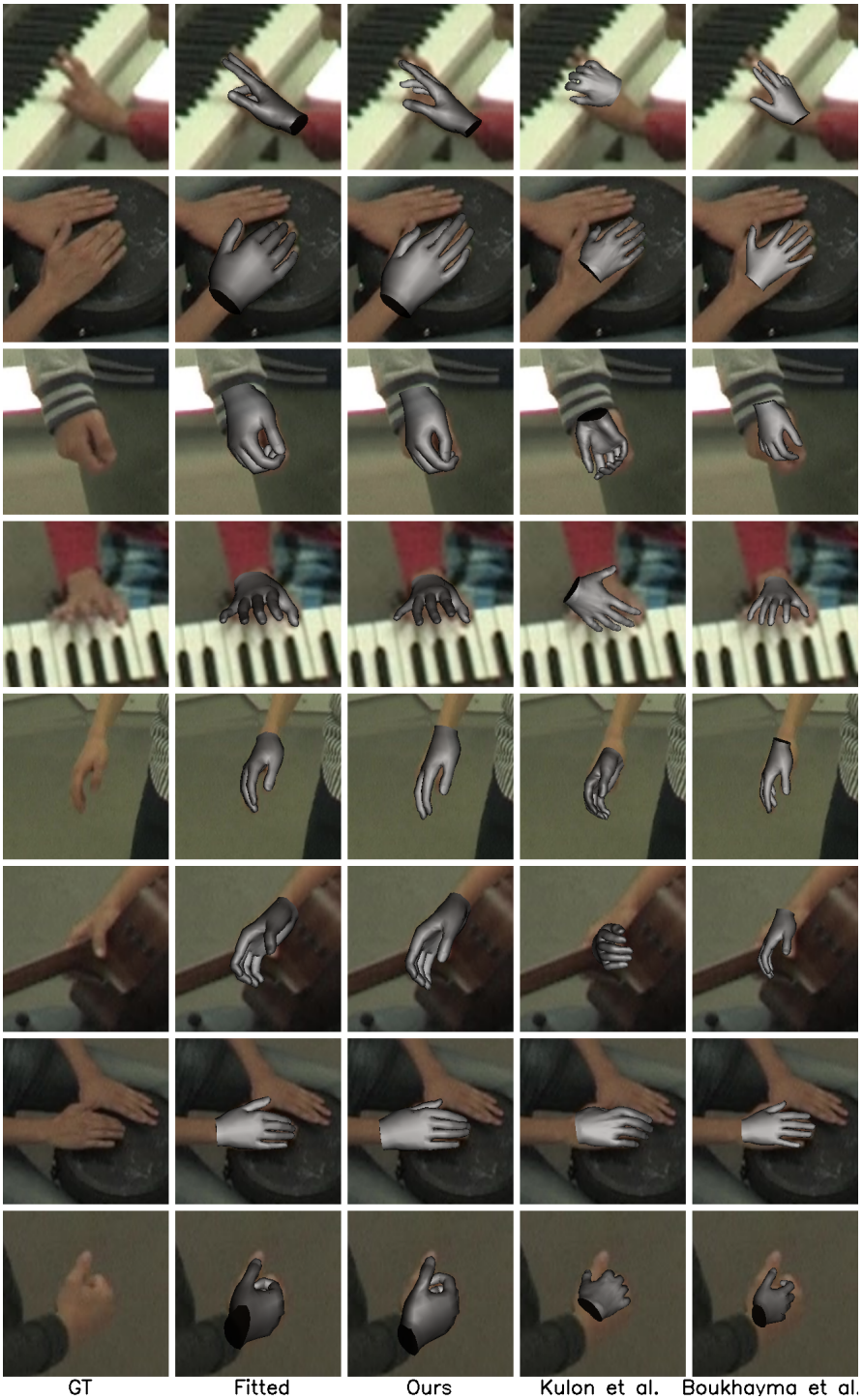


Figure 5.5: Qualitative evaluation results of the fitted hand mesh (the second column), our hand reconstruction approach (the third column), Kulon *et al.* [20] (the fourth column), and Boukhayma *et al.* [21] (the fifth column).

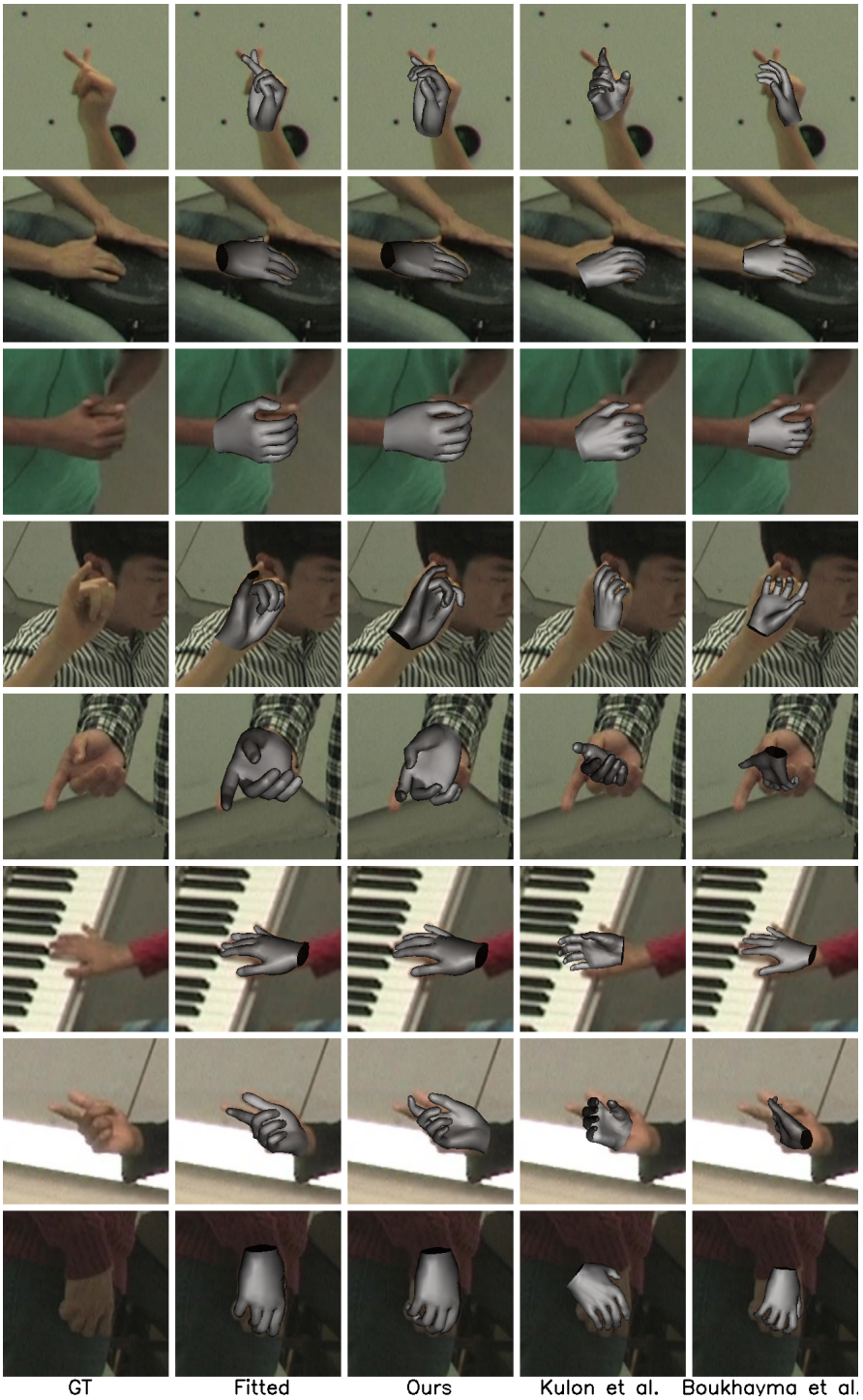


Figure 5.6: Qualitative evaluation results of the fitted hand mesh (the second column), our hand reconstruction approach (the third column), Kulon *et al.* [20] (the fourth column), and Boukhayma *et al.* [21] (the fifth column).

error distance is below a specific threshold;

- AUC: the area under the curve (AUC) on the 2D/3D PCK for different error thresholds.

We compare our approach with the state-of-the-art 3D hand mesh estimation methods on the Panoptic Dataset. Specifically, we report the results from the approaches of Boukhayma *et al.* [21] and Kulon *et al.* [20]. We use the implementations provided by the authors and fine-tune the model of Boukhayma *et al.* [21] on the Panoptic Dataset. The approach of Kulon *et al.* [20] has already been trained with the Panoptic Dataset so we use the model provided by the authors directly for evaluation.

The 2D PCK and 3D PCK curves over different error thresholds are presented in Figure 5.7. Because we are using weak perspective camera for projection, we evaluate the 3D PCK after depth alignment in the camera coordinates. Our method outperforms the two state-of-the-art methods over all the evaluation metrics.

5.5.3 Qualitative Evaluation of 3D Hand Mesh Estimation

The qualitative results of 3D hand mesh reconstruction appear in Figures 5.4 to 5.6.

5.5.4 Ablation Study

Ablation Study of Loss terms: We evaluate the impact of the embedding loss $L_{embedding}$ and the joint loss L_{joint} in the fully supervised training. As shown in Figure 5.9, the model trained with only the embedding loss might make a large

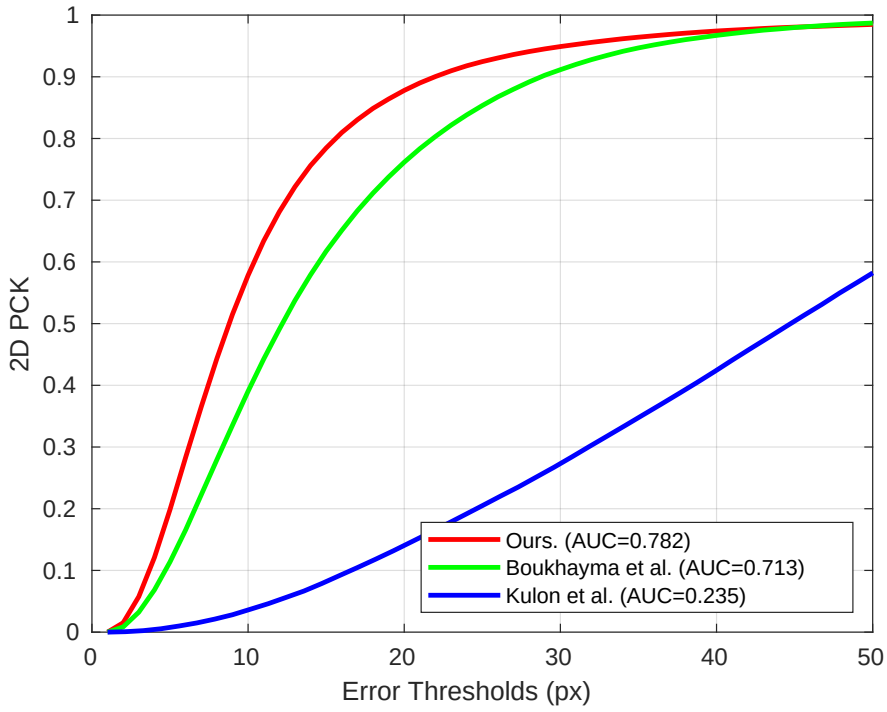


Figure 5.7: 2D PCK our hand reconstruction approach, Kulon *et al.* [20], and Boukhayma *et al.* [21]. Our method outperforms the other methods in AUC.

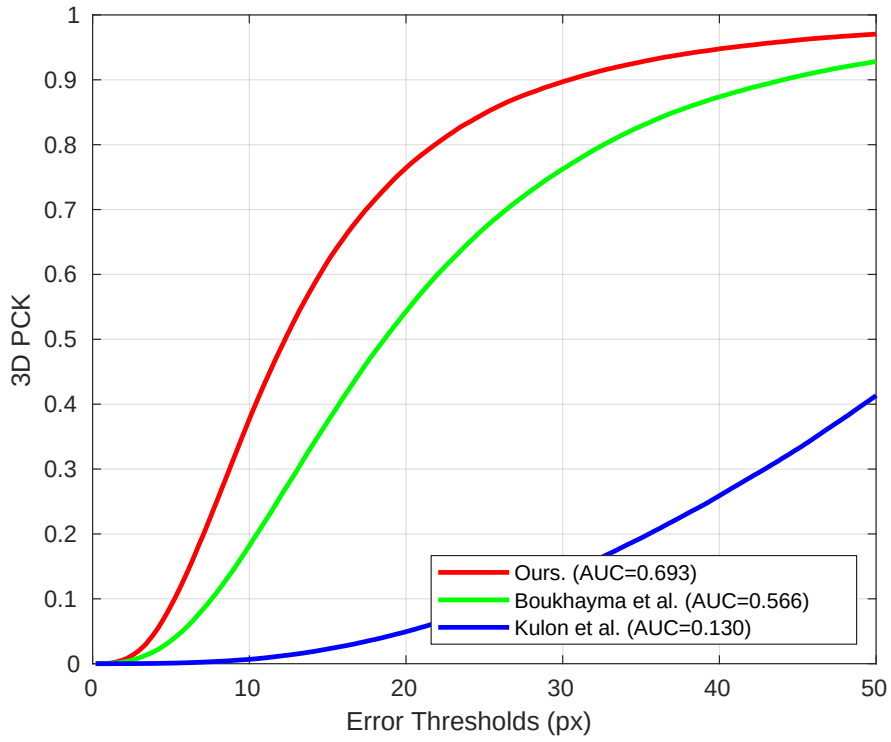


Figure 5.8: 3D PCK our hand reconstruction approach, Kulon *et al.* [20], and Boukhayma *et al.* [21]. Our method outperforms the other methods in AUC.

prediction error by wrongly predicting a parameter such as rotation or translation. The model trained with only the joint loss might generate a twisted (physically implausible) hand to minimize the joint error. The PCK curve presented in Figure 5.10 also shows that the model trained with the sum of joint loss and parameter loss as the objective achieves the best performance.

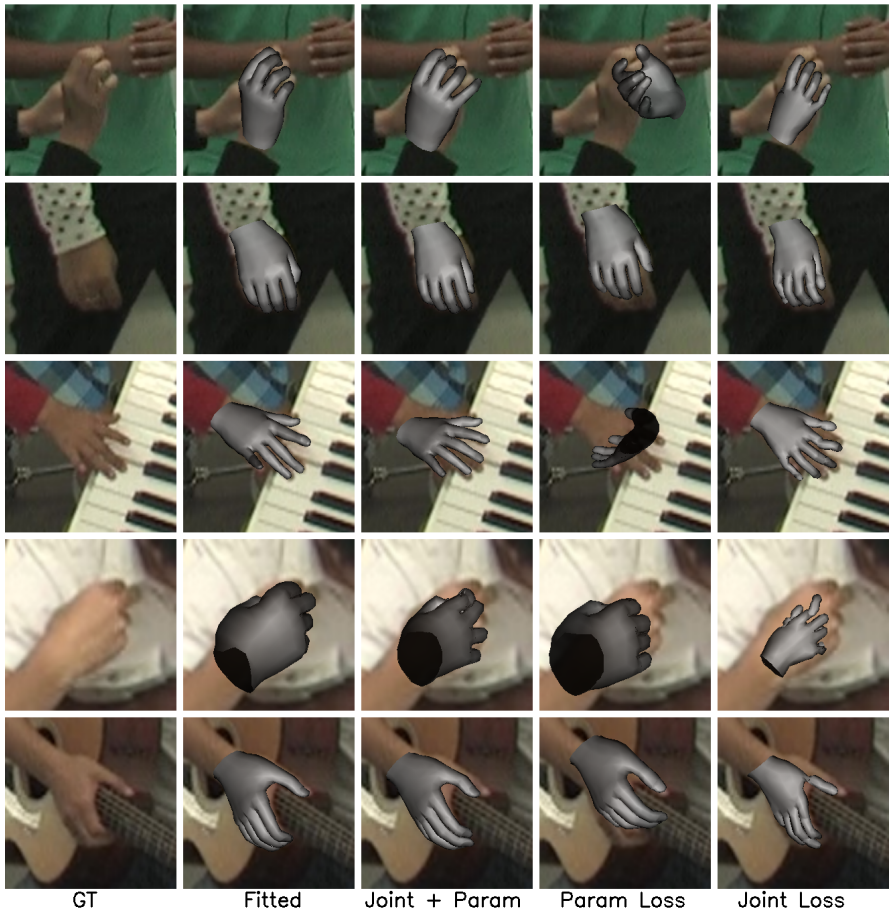


Figure 5.9: Qualitative evaluation results of the fitted hand mesh (the second column), our hand reconstruction approach (the third column) with the sum of joint loss and parameter loss as the objective, our hand reconstruction approach (the fourth column) with the parameter loss as the objective, and our hand reconstruction approach (the fifth column) with the joint loss as the objective.

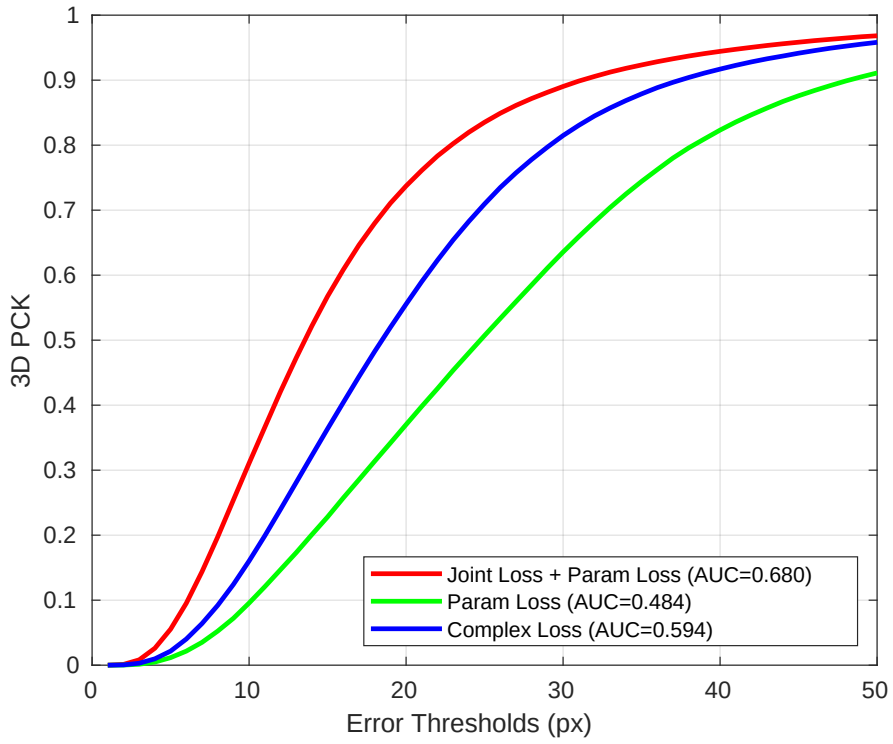


Figure 5.10: 3D PCK of our hand reconstruction approach with the sum of joint loss and parameter loss as the objective, our hand reconstruction approach with the parameter loss as the objective, and our hand reconstruction approach with the joint loss as the objective.

Ablation Study of Input Type: We evaluate the impact of the RGB image and the heatmaps in the fully supervised training. We train the autoencoder for 450 epochs with RGB image and heatmaps, with only RGB images, and with only heatmaps. As shown in Figure 5.11, the model trained with only the RGB image and the model trained with only the heatmaps predicts wrong pose or hand rotation. The PCK curve presented in Figure 5.12 shows that the model trained with the RGB images and the heatmaps achieves the best performance in the hand mesh

estimation, which indicates that both inputs contribute to the improvement.

Figure 5.11: Qualitative evaluation results of the fitted hand mesh (the second column), our hand reconstruction approach (the third column) with the RGB image and heatmaps as inputs, our hand reconstruction approach (the fourth column) with the RGB image as input, and our hand reconstruction approach (the fifth column) with heatmaps as input.

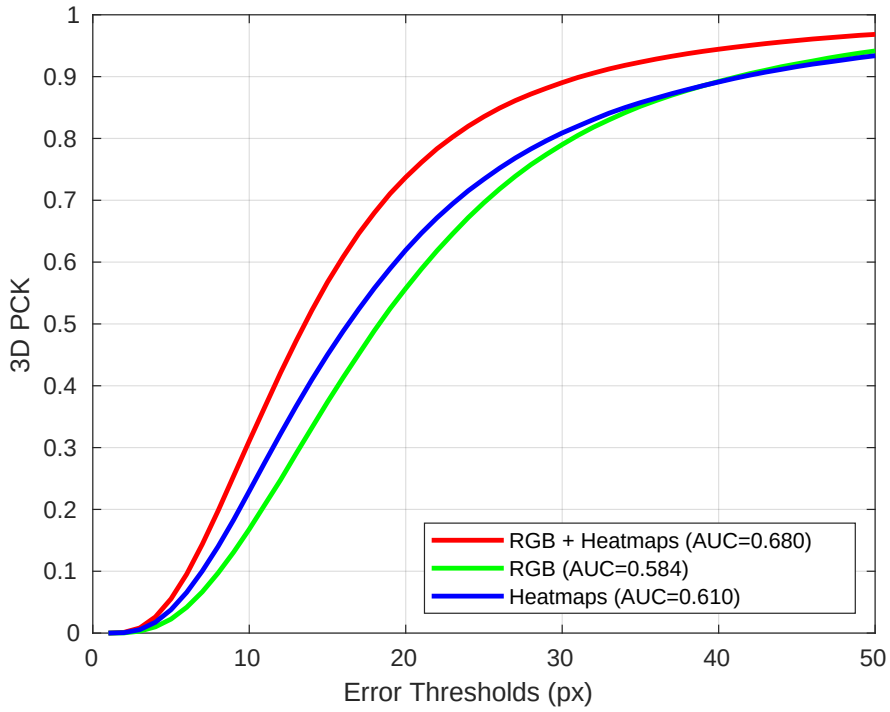


Figure 5.12: 3D PCK of our hand reconstruction approach with the RGB image and heatmaps as inputs, our hand reconstruction approach with the RGB image as input, and our hand reconstruction approach with heatmaps as input.

Chapter 6: Conclusion and Future Work

In this dissertation, I have presented my research to enhance visual and and gestural fidelity for effective visual environments.

In *Kernel Foveated Rendering* [30], I have presented the kernel log-polar mapping model and conducted user studies for finding the best parameters, as well as a GPU-based implementation and quantitative evaluation of the kernel foveated rendering pipeline. With high frame rates, the KFR pipeline allows rendering more complex shaders (*e.g.,* real-time global illumination and physically-based rendering [129]) in real time, thus bringing higher power efficiency and better user experience for 3D games and other interactive visual computing applications.

Even though I have devised an efficient and effective foveated rendering pipeline, my system is not without some limitations.

Foveation Parameters As discussed in Hsu *et al.* [130], the perceived quality of foveated rendering systems is highly dependent on the user and the scene. As the initial step towards kernel foveated rendering for 3D graphics, the user study in this project is only designed for selected static scenes. The foveation parameters may vary in dynamic scenes. Exploring the relationship between user demographic (*e.g.*, pupil size, contrast sensitivity, vision condition, and diopter), perception time [26]

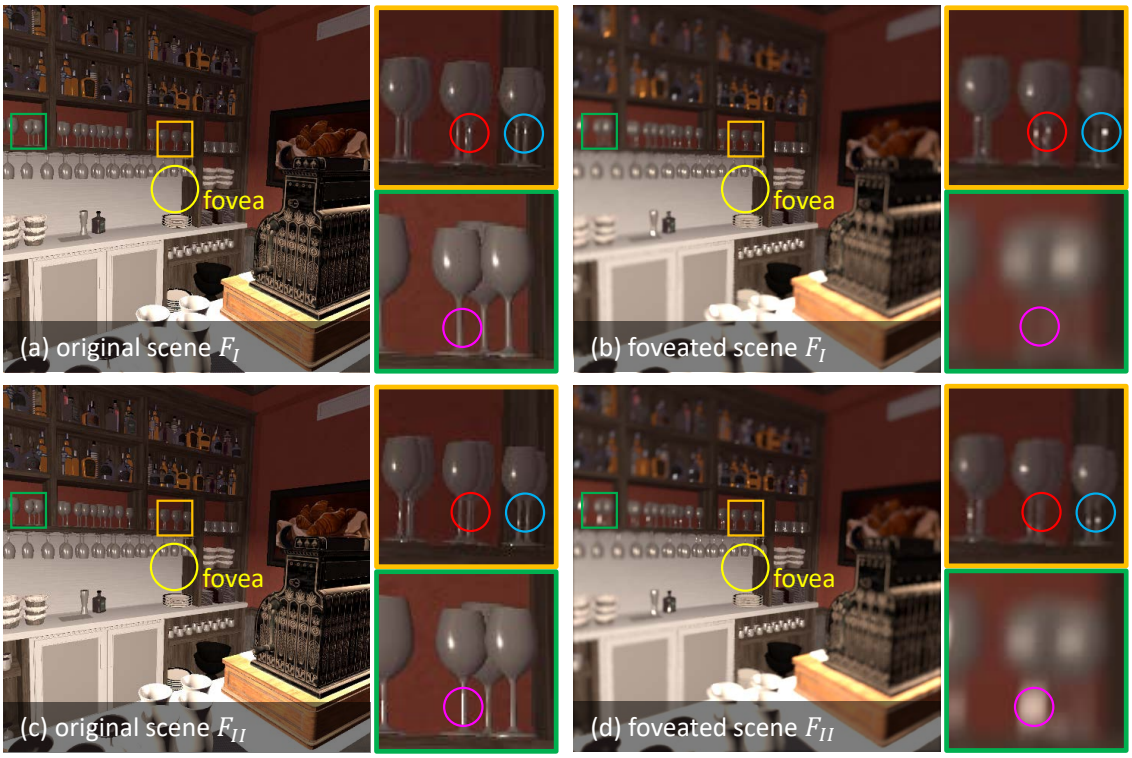


Figure 6.1: Temporal flickering issue. The original scene and the foveated scene of two consecutive frames (F_I and F_{II}). In F_I, the specular reflection in the original scene as shown in the red and blue circles in the zoomed-in view of (a) are amplified in the foveated scene as shown in the zoomed-in view of (b). In the next frame F_{II}, the specular reflection in the original scene as shown in the pink circle in the zoomed-in view of (c) is amplified in the foveated scene as shown in the zoomed-in view of (d).

and display-dependent parameters of KFR is a potential future direction.

Temporal Flickering In the post-processing stage, I have applied TAA to tackle the temporal flickering problem. However, in fly-through of the scene with glossy objects, I notice that view-dependent specular reflection changes before and after applying KFR. As shown in Figure 6.1, foveation amplifies the specular reflection regions, and makes the specular highlights flicker more.

Other Mapping Algorithms and Kernel Functions KFR makes intuitive sense as the log-polar mapping has an initial resolution proportional to e^{-r}, and the kernel functions can fine tune this mapping. My choice of kernel functions is not unique; other mapping algorithms with different kernel functions could provide a better mapping to the human vision system. As shown in Figure 6.2, Koskela *et al.* [131] implement a path traced frame in Visual-Polar space. The visual polar space performs better than log-polar space in reducing distracting artifacts.

In *3D-kernel Foveated Rendering for Light Fields* [23], I have presented a novel approach to accelerate the interactive visualization of high-resolution light fields. We conduct user studies to determine the optimal foveation parameters to validate the 3D-KFR pipeline in practice. According to the quantitative experiments, our methods accelerate the rendering process of large-scale, high-resolution light fields by a factor of up to 7.28× at the resolution of $25 \times 25 \times 1024 \times 1024$.

Our algorithm is effective in rendering high resolution light fields using foveation for virtual reality HMD with low latency, low power consumption and minimal

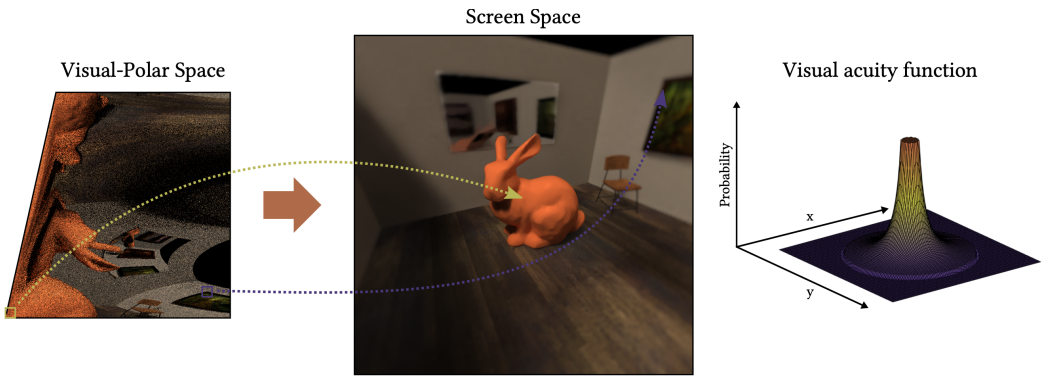


Figure 6.2: Illustration of a path traced frame in Visual-Polar space, the denoised result transformed into Cartesian screen space, and the distribution of the path tracing samples in screen space. Path tracing and denoising in Visual-Polar space makes both 2.5× faster.

perceptual differences. With the increase of VR headset resolution and the growth of the VR market, we envision that 3D-KFR may inspire further research in the foveated rendering of high-resolution light fields.

There are several possibilities to further improve our algorithm.

Foveation Parameters Our choice of the relationship between σ_0, σ_1, and σ_2 is not unique. Other sigma arrays may provide a higher speedup. However, the trade-off between rendering quality and foveation parameter σ always exists. It is desirable to further explore the relationship between rendering quality and σ.

In *Eye-dominance-guided Foveated Rendering* [31], I have presented the EFR pipeline, which achieves a significant speed-up by rendering the scene in the dominant eye with a lower foveation level (higher detail) and rendering the scene in the non-

dominant eye with a higher foveation level (lower detail). This technique takes advantage of the ocular dominance property of the human visual system, and leverages the difference in acuity and sensitivity between the dominant eye and the non-dominant eye. Our approach can be easily integrated into the current rasterization rendering pipeline for head-mounted displays. We envision that EFR would be also beneficial to data streaming for networked VR/AR applications such as Montage4D [132], Geollery [133, 134], AR surgery [135], and memory palaces [136] by reducing the bandwidth requirements.

Temporal Artifacts One of the grand challenges in foveated rendering is handling artifacts due to temporal aliasing of moving objects [137], phase-aligned aliasing [88], and saliency-map based aliasing [138]. Since the eye-dominance-guided foveated rendering relies on different levels of foveation for the two eyes, such challenges are likely to be even greater. We plan to study and address these challenges in future.

Personalized VR Rendering Ocular dominance studies [139] indicate that 70% of the population is right-eye dominant and 29% is left-eye dominant. Thus, we expect that most users stand to benefit from eye-dominance-guided foveated rendering. In terms of personalized VR rendering, prior art has investigated how to personalize spatial audio for virtual environments using head-related transfer functions based on the ears' shape [140]. Further research may investigate how to enhance the visual experience of a user based on the eye prescription.

Further Leveraging Human Perception An important argument in the study of visual direction is that there is a center or origin for judgments of visual direction called cyclopean eye [141]. Elbaum *et al.* [142] demonstrates that tracking accuracy is better with the cyclopean eye than with the dominant and non-dominant eye. Xia and Peli [143] propose a perceptual space model for virtual reality content based on the gaze point of the cyclopean eye. How the human visual system integrates the input from the two eyes into a cyclopean vision and how virtual reality in general, and foveated rendering in particular, could leverage it to improve visual quality and efficiency is deeply intriguing. We plan to delve into exploring how the foveated rendering system could be integrated with the cyclopean eye to further improve the immersive viewing experience and enhance the interaction accuracy between HMD and users.

In *Hand Reconstruction from RGB Images*, I have presented an end-to-end convolutional neural network that predicts 3D hand shape and pose from a single RGB image. The proposed research solves the problem of sparse training annotation by fitting a 3D deformation model to $74,715$ 3D possible joint conformations and improves the quality of estimation. We envision that the proposed approach could be used in multiple application scenarios in the field of human-computer interaction and virtual and augmented reality.There are improvements that we can make to enhance the user's immersive experience.

Texture Reconstruction The first research direction is to predict the hand texture for the estimated hand mesh. We plan to use a captured hand texture as the

template and use another neural network to predict the surface texture parameters such as the hand color and roughness. We could use the texture template and the surface texture parameters to synthesize a texture that will be attached to the estimated hand for visualization.

Hand Interaction The second challenge is to tackle potential problems like the interaction between the two hands. Mueller *et al.* [144] focuses on the hand interaction captured from depth cameras. Because depth data is not available in most VR headsets, it will be desirable to focus on handling inter-hand and intra-hand collisions with only a sequence of RGB images as input.

Finally, it will be interesting to develop a real-time VR application that accepts the video from the inside-out camera as the input, predicts the hand mesh from the video, and visualizes the predicted hand mesh.

Bibliography

[1] Christine A Curcio, Kenneth R Sloan, Robert E Kalina, and Anita E Hendrickson. Human photoreceptor topography. *Journal of comparative neurology*, 292(4):497–523, 1990.

[2] Christine A Curcio and Kimberly A Allen. Topography of ganglion cells in human retina. *Journal of comparative Neurology*, 300(1):5–25, 1990.

[3] Philip Kortum and Wilson S Geisler. Implementation of a foveated image coding system for image bandwidth reduction. In *Electronic Imaging: Science & Technology*, pages 350–360. International Society for Optics and Photonics, 1996.

[4] P. Lungaro, R. Sjöberg, A. J. F. Valero, A. Mittal, and K. Tollmar. Gaze-aware streaming solutions for the next generation of mobile vr experiences. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1535–1544, April 2018.

[5] S. Firdose, P. Lunqaro, and K. Tollmar. Demonstration of Gaze-Aware Video Streaming Solutions for Mobile VR. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 749–750, March 2018.

[6] Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. Foveated 3d graphics. *ACM Trans. Graph.*, 31(6):164:1–164:10, November 2012.

[7] K. Vaidyanathan, M. Salvi, R. Toth, T. Foley, T. Akenine-Möller, J. Nilsson, J. Munkberg, J. Hasselgren, M. Sugihara, P. Clarberg, T. Janczak, and A. Lefohn. Coarse pixel shading. In *Proceedings of High Performance Graphics*, HPG '14, pages 9–18, Aire-la-Ville, Switzerland, Switzerland, 2014. Eurographics Association.

[8] Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Trans. Graph.*, 35(6):179:1–179:12, November 2016.

[9] Anjul Patney, Joohwan Kim, Marco Salvi, Anton Kaplanyan, Chris Wyman, Nir Benty, Aaron Lefohn, and David Luebke. Perceptually-based foveated virtual reality. In *ACM SIGGRAPH 2016 Emerging Technologies*, SIGGRAPH '16, pages 17:1–17:2, New York, NY, USA, 2016. ACM.

[10] Petrik Clarberg, Robert Toth, Jon Hasselgren, Jim Nilsson, and Tomas Akenine-Möller. Amfs: adaptive multi-frequency shading for future graphics processors. *ACM Trans. Graph.*, 33(4):141:1–141:12, July 2014.

[11] Yong He, Yan Gu, and Kayvon Fatahalian. Extending the graphics pipeline with adaptive, multi-rate shading. *ACM Trans. Graph.*, 33(4):142:1–142:12, July 2014.

[12] Nicholas T. Swafford, José A. Iglesias-Guitian, Charalampos Koniaris, Bochang Moon, Darren Cosker, and Kenny Mitchell. User, metric, and computational evaluation of foveated rendering methods. In *Proceedings of the ACM Symposium on Applied Perception*, SAP '16, pages 7–14, New York, NY, USA, 2016. ACM.

[13] Michael Stengel, Steve Grogorick, Martin Eisemann, and Marcus Magnor. Adaptive image-space sampling for gaze-contingent real-time rendering. *Computer Graphics Forum*, 35(4):129–139, 2016.

[14] Okan Tarhan Tursun, Elena Arabadzhiyska-Koleva, Marek Wernikowski, Radosław Mantiuk, Hans-Peter Seidel, Karol Myszkowski, and Piotr Didyk. Luminance-contrast-aware foveated rendering. *ACM Trans. Graph.*, 38(4):98:1–98:14, July 2019.

[15] Cheuk Yiu Ip, M. Adil Yalçin, David Luebke, and Amitabh Varshney. Pixelpie: maximal poisson-disk sampling with rasterization. In *Proceedings of the 5th High-Performance Graphics Conference*, HPG '13, pages 17–26, New York, NY, USA, 2013. ACM.

[16] Marc Levoy and Pat Hanrahan. Light Field Rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 31–42, New York, NY, USA, 1996. ACM.

[17] Qi Sun, Fu-Chung Huang, Joohwan Kim, Li-Yi Wei, David Luebke, and Arie Kaufman. Perceptually-guided foveation for light field displays. *ACM Trans. Graph.*, 36(6):192:1–192:13, November 2017.

[18] Morgan McGuire. Computer graphics archive, July 2017. `https://casual-effects.com/data`.

[19] Amazon Lumberyard. Amazon lumberyard bistro, open research content archive (orca), July 2017.

[20] Dominik Kulon, Haoyang Wang, Riza Alp Güler, Michael M. Bronstein, and Stefanos Zafeiriou. Single image 3d hand reconstruction with mesh convolutions. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2019.

[21] Adnane Boukhayma, Rodrigo de Bem, and Philip H.S. Torr. 3d hand shape and pose from images in the wild. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[22] G. Denes, K. Maruszczyk, G. Ash, and R. K. Mantiuk. Temporal Resolution Multiplexing: Exploiting the limitations of spatio-temporal vision for more efficient VR rendering. *IEEE Transactions on Visualization and Computer Graphics*, 25(5):2072–2082, May 2019.

[23] Xiaoxu Meng, Ruofei Du, Matthias Zwicker, and Amitabh Varshney. Kernel Foveated Rendering. *Proc. ACM Comput. Graph. Interact. Tech*, 1(1):5:1–5:20, May 2018.

[24] Marc Levoy and Ross Whitaker. Gaze-directed volume rendering. In *Proceedings of the 1990 Symposium on Interactive 3D Graphics*, I3D '90, pages 217–223, New York, NY, USA, 1990. ACM.

[25] Hans Strasburger, Ingo Rentschler, and Martin Jüttner. Peripheral vision and pattern recognition: a review. *Journal of vision*, 11(5):13–13, 2011.

[26] Xuetong Sun and Amitabh Varshney. Investigating Perception Time in the Far Peripheral Vision for Virtual and Augmented Reality. In *ACM Symposium on Applied Perception (SAP)*, Perception. ACM, Aug 2018.

[27] E. L. Schwartz. Anatomical and physiological correlates of visual computation from striate to infero-temporal cortex. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(2):257–271, March 1984.

[28] H. Araujo and J. M. Dias. An introduction to the log-polar mapping [image sampling]. In *Proceedings II Workshop on Cybernetic Vision*, pages 139–144, Dec 1996.

[29] Marco Antonelli, Francisco D. Igual, Francisco Ramos, and V. Javier Traver. Speeding up the log-polar transform with inexpensive parallel hardware: graphics units and multi-core architectures. *J. Real-Time Image Process.*, 10(3):533–550, September 2015.

[30] Xiaoxu Meng, Ruofei Du, Matthias Zwicker, and Amitabh Varshney. Kernel foveated rendering. *Proc. ACM Comput. Graph. Interact. Tech.*, 1(1):5:1–5:20, July 2018.

[31] X. Meng, R. Du, and A. Varshney. Eye-dominance-guided foveated rendering. *IEEE Transactions on Visualization and Computer Graphics*, 26(5):1972–1980, 2020.

[32] Shawn Hargreaves and Mark Harris. Deferred shading. In *Game Developers Conference*, volume 2, page 31, 2004.

[33] Jerome F Duluk Jr, Richard E Hessel, Vaughn T Arnold, Jack Benkual, Joseph P Bratt, George Cuan, Stephen L Dodgen, Emerson S Fang, Zhaoyu Gong, Thomas Y Ho, et al. Deferred shading graphics pipeline processor having advanced features, April 6 2004. US Patent 6,717,576.

[34] Peter J Burt. Smart sensing within a pyramid vision machine. *Proceedings of the IEEE*, 76(8):1006–1015, 1988.

[35] Jerome M Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41(12):3445–3462, 1993.

[36] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):243–250, Jun 1996.

[37] Ee-Chien Chang, Stéphane Mallat, and Chee Yap. Wavelet foveation. *Applied and Computational Harmonic Analysis*, 9(3):312–335, 2000.

[38] Zhou Wang and Alan C Bovik. Embedded foveation image coding. *IEEE Transactions on Image Processing*, 10(10):1397–1410, 2001.

[39] C. Papadopoulos and A. E. Kaufman. Acuity-driven gigapixel visualization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2886–2895, Dec 2013.

[40] T. H. Reeves and J. A. Robinson. Adaptive foveation of mpeg video. In *Proceedings of the Fourth ACM International Conference on Multimedia*, MULTIMEDIA '96, pages 231–241, New York, NY, USA, 1996. ACM.

[41] Lee and Sanghoon. *Foveated video compression and visual communications over wireless and wireline networks*. PhD thesis, Dept. of ECE, University of Texas at Austin, 2000.

[42] Zhou Wang and Alan C Bovik. Foveated image and video coding. In *Digitial Video Image Quality and Perceptual Coding*, pages 1–28. 2005.

[43] Sanghoon Lee, M. S. Pattichis, and A. C. Bovik. Foveated video compression with optimal rate control. *IEEE Transactions on Image Processing*, 10(7):977–992, Jul 2001.

[44] Sanghoon Lee, M. S. Pattichis, and A. C. Bovik. Foveated video quality assessment. *IEEE Transactions on Multimedia*, 4(1):129–132, Mar 2002.

[45] H. R. Sheikh, S. Liu, Z. Wang, and A. C. Bovik. Foveated multipoint videoconferencing at low bit rates. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages II–2069–II–2072, May 2002.

[46] Hamid R. Sheikh, Brian L. Evans, and Alan C. Bovik. Real-time foveation techniques for low bit rate video coding. *Real-Time Imaging*, 9(1):27–40, February 2003.

[47] Anton Kaplanyan, Anton Sochenov, Thomas Leimkühler, Mikhail Okunev, Todd Goodall, and Gizem Rufo. Deepfovea: Neural reconstruction for foveated rendering and video compression using learned natural video statistics. In *ACM SIGGRAPH 2019 Talks*, SIGGRAPH '19, pages 58:1–58:2, New York, NY, USA, 2019. ACM.

[48] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

[49] M. Weier, M. Stengel, T. Roth, P. Didyk, E. Eisemann, M. Eisemann, S. Grogorick, A. Hinkenjann, E. Kruijff, M. Magnor, K. Myszkowski, and P. Slusallek. Perception-driven accelerated rendering. *Comput. Graph. Forum*, 36(2):611–643, May 2017.

[50] Toshikazu Ohshima, Hiroyuki Yamamoto, and Hideyuki Tamura. Gaze-directed adaptive rendering for interacting with virtual space. In *Proceedings of the 1996 Virtual Reality Annual International Symposium (VRAIS 96)*, VRAIS '96, pages 103–110, 267, Washington, DC, USA, 1996. IEEE Computer Society.

[51] Hugues Hoppe. Smooth view-dependent level-of-detail control and its application to terrain rendering. In *Proceedings of the Conference on Visualization '98*, VIS '98, pages 35–42, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[52] L. Hu, P. V. Sander, and H. Hoppe. Parallel view-dependent level-of-detail control. *IEEE Transactions on Visualization and Computer Graphics*, 16(5):718–728, Sept 2010.

[53] Jonathan Ragan-Kelley, Jaakko Lehtinen, Jiawen Chen, Michael Doggett, and Frédo Durand. Decoupled sampling for graphics pipelines. *ACM Trans. Graph.*, 30(3):17:1–17:17, May 2011.

[54] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 43–54, New York, NY, USA, 1996. ACM.

[55] Jonghyun Kim, Youngmo Jeong, Michael Stengel, Kaan Akşit, Rachel Albert, Ben Boudaoud, Trey Greer, Joohwan Kim, Ward Lopes, Zander Majercik, Peter Shirley, Josef Spjut, Morgan McGuire, and David Luebke. Foveated AR: Dynamically-foveated Augmented Reality Display. *ACM Trans. Graph.*, 38(4):99:1–99:15, July 2019.

[56] B Karis. High-quality temporal supersampling. *Advances in Real-Time Rendering in Games, SIGGRAPH Courses*, 1:1–55, 2014.

[57] Cyril Crassin, Morgan McGuire, Kayvon Fatahalian, and Aaron Lefohn. Aggregate g-buffer anti-aliasing. In *Proceedings of the 19th Symposium on Interactive 3D Graphics and Games*, I3D '15, pages 109–119, New York, NY, USA, 2015. ACM.

[58] Matthäus G. Chajdas, Morgan McGuire, and David Luebke. Subpixel reconstruction antialiasing for deferred shading. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 15–22 PAGE@7, New York, NY, USA, 2011. ACM.

[59] T Pengo, A Muñoz-Barrutia, and C Ortiz-de solórzano. Halton sampling for autofocus. *Journal of Microscopy*, 235(1):50–58, 2009.

[60] Kashinath D Patil. Cochran's q test: Exact distribution. *Journal of the American Statistical Association*, 70(349):186–189, 1975.

[61] Margarita Vinnikov and Robert S Allison. Gaze-contingent depth of field in realistic scenes: The user experience. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 119–126. ACM, 2014.

[62] Nir Benty, Kai-Hwa Yao, Tim Foley, Anton S. Kaplanyan, Conor Lavelle, Chris Wyman, and Ashwin Vijay. The Falcor rendering framework, 07 2017.

[63] Frank W Weymouth. Visual sensory units and the minimal angle of resolution. *American Journal of Ophthalmology*, 46(1):102–113, 1958.

[64] Chang Ha Lee, Amitabh Varshney, and David Jacobs. Mesh saliency. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, 24(3):659 – 666, August 2005.

[65] Youngmin Kim, Amitabh Varshney, David Jacobs, and Francois Guimbretere. Mesh saliency and human eye fixations. *ACM Transactions on Applied Perception*, 7(2):1 – 13, 2010.

[66] Hsueh-Chien Cheng, Antonio Cardone, Eric Krokos, Bogdan Stoica, Alan Faden, and Amitabh Varshney. Deep-Learning-Assisted Visualization for Live-Cell Images. In *Proceedings of 2017 IEEE International Conference on Image Processing*, ICIP. IEEE, September 2017.

[67] Y. Wan, H. Otsuna, C. Chien, and C. Hansen. An Interactive Visualization Tool for Multi-Channel Confocal Microscopy Data in Neurobiology Research. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1489–1496, Nov 2009.

[68] M. Hadwiger, J. Beyer, W. Jeong, and H. Pfister. Interactive Volume Exploration of Petascale Microscopy Data Streams Using a Visualization-Driven Virtual Memory Approach. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2285–2294, Dec 2012.

[69] K. Mosaliganti, L. Cooper, R. Sharp, R. Machiraju, G. Leone, K. Huang, and J. Saltz. Reconstruction of Cellular Biological Structures From Optical Microscopy Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):863–876, July 2008.

[70] Hsueh-Chien Cheng, Antonio Cardone, Somay Jain, Eric Krokos, Kedar Narayan, Sriram Subramaniam, and Amitabh Varshney. Deep-learning-assisted Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–14, January 2018.

[71] Hsueh-Chien Cheng, Antonio Cardone, and Amitabh Varshney. Volume Segmentation Using Convolutional Neural Networks With Limited Training Data. In *Proceedings of 2017 IEEE International Conference on Image Processing*, ICIP. IEEE, September 2017.

[72] Marc Levoy, Ren Ng, Andrew Adams, Matthew Footer, and Mark Horowitz. Light Field Microscopy. *ACM Trans. Graph*, 25(3):924–934, 2006.

[73] Robert Prevedel, Young-Gyu Yoon, Maximilian Hoffmann, Nikita Pak, Gordon Wetzstein, Saul Kato, Tina Schrödel, Ramesh Raskar, Manuel Zimmer, Edward S Boyden, et al. Simultaneous Whole-Animal 3D Imaging of Neuronal Activity Using Light-Field Microscopy. *Nature Methods*, 11(7):727, 2014.

[74] Jin-Xiang Chai, Xin Tong, Shing-Chow Chan, and Heung-Yeung Shum. Plenoptic Sampling. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 307–318, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[75] Ren Ng. Fourier Slice Photography. *ACM Trans. Graph*, 24(3):735–744, 2005.

[76] Douglas Lanman and David Luebke. Near-Eye Light Field Displays. In *ACM SIGGRAPH 2013 Emerging Technologies*, SIGGRAPH '13, pages 11:1–11:1, New York, NY, USA, 2013. ACM.

[77] Fu-Chung Huang, Kevin Chen, and Gordon Wetzstein. The Light Field Stereoscope: Immersive Computer Graphics Via Factored Near-Eye Light Field Displays With Focus Cues. *ACM Trans. Graph*, 34(4):60:1–60:12, 2015.

[78] J. Zhang, Z. Fan, D. Sun, and H. Liao. Unified Mathematical Model for Multilayer-Multiframe Compressive Light Field Displays Using LCDs. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2018.

[79] S. Lee, J. Cho, B. Lee, Y. Jo, C. Jang, D. Kim, and B. Lee. Foveated Retinal Optimization for See-Through Near-Eye Multi-Layer Displays. *IEEE Access*, 6:2170–2180, 2018.

[80] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019.

[81] Ronald S. Weinstein, Michael R. Descour, Chen Liang, Gail Barker, Katherine M. Scott, Lynne Richter, Elizabeth A. Krupinski, Achyut K. Bhattacharyya, John R. Davis, Anna R. Graham, Margaret Rennels, William C. Russum, James F. Goodall, Pixuan Zhou, Artur G. Olszak, Bruce H. Williams, James C. Wyant, and Peter H. Bartels. An Array Microscope for Ultrarapid Virtual Slide Processing and Telepathology. Design, Fabrication, and Validation Study. *Human Pathology*, 35(11):1303 – 1314, 2004.

[82] Brian Wilt, Laurie Burns, Eric Tatt Wei Ho, Kunal Ghosh, Eran Mukamel, and Mark Schnitzer. Advances in Light Microscopy for Neuroscience. *Annual Review of Neuroscience*, pages 435–506, 9.

[83] Robert Prevedel, Young-Gyu Yoon, Maximilian Hoffmann, Nikita Pak, Gordon Wetzstein, Saul Kato, Tina Schrödel, Ramesh Raskar, Manuel Zimmer, Edward S Boyden, and Alipasha Vaziri. Simultaneous Whole-Animal 3D Imaging of Neuronal

Activity Using Light-Field Microscopy. *Nature Methods*, 11:727 – 730, 05/18/2014 2014.

[84] B. Sheng, P. Li, Y. Jin, P. Tan, and T. Lee. Intrinsic image decomposition with step and drift shading separation. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2018.

[85] Q. Chen and V. Koltun. A simple model for intrinsic image decomposition with depth cues. In *2013 IEEE International Conference on Computer Vision*, pages 241–248, Dec 2013.

[86] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

[87] W. Feng, Y. Yang, L. Wan, and C. Yu. Tone-mapped mean-shift based environment map sampling. *IEEE Transactions on Visualization and Computer Graphics*, 22(9):2187–2199, Sep. 2016.

[88] E. Turner, H. Jiang, D. Saint-Macary, and B. Bastani. Phase-Aligned Foveated Rendering for Virtual Reality Headsets. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1–2, Los Alamitos, CA, USA, mar 2018. IEEE Computer Society.

[89] Piotr Didyk, Elmar Eisemann, Tobias Ritschel, Karol Myszkowski, and Hans-Peter Seidel. Perceptually-motivated Real-time Temporal Upsampling of 3D Content for High-refresh-rate Displays. *Computer Graphics Forum*, 29(2):713–722, 2010.

[90] Clare Kathleen Porac and Stanley Coren. The dominant eye. *Psychological Bulletin*, 83(5):880–897, 9 1976.

[91] Einat Shneor and Shaul Hochstein. Eye dominance effects in feature search. *Vision Research*, 46(25):4258 – 4269, 2006.

[92] Belkıs Koçtekin, Nimet Ünay Gündoğan, Ayş Gül Koçak Altıntaş, and Ayşe Canan Yazıcı. Relation of eye dominancy with color vision discrimination performance ability in normal subjects. *International journal of ophthalmology*, 6(5):733, 2013.

[93] I. C. McManus, Clare Kathleen Porac, M. P. Bryden, and R. Boucher. Eye-dominance, Writing Hand, and Throwing Hand. *Laterality: Asymmetries of Body, Brain and Cognition*, 4(2):173–192, 1 1999.

[94] Ayame Oishi, Shozo Tobimatsu, Kenji Arakawa, Takayuki Taniwaki, and Jun ichi Kira. Ocular dominancy in conjugate eye movements at reading distance. *Neuroscience Research*, 52(3):263 – 268, 2005.

[95] Romain Chaumillon, Jean Blouin, and Alain Guillaume. Eye dominance influences triggering action: The Poffenberger paradigm revisited. *Cortex*, 58:86 – 98, 2014.

[96] Carlo A Marzi. The poffenberger paradigm: a first, simple, behavioural tool to study interhemispheric transmission in humans. *Brain Research Bulletin*, 50(5):421 – 422, 1999.

[97] Heidi L. Roth, Andrea N. Lora, and Kenneth M. Heilman. Effects of monocular viewing and eye dominance on spatial attention. *Brain*, 125(9):2023–2035, 09 2002.

[98] Michael R. Stoline. The Status of Multiple Comparisons: Simultaneous Estimation of all Pairwise Comparisons in One-Way ANOVA Designs. *The American Statistician*, 35(3):134–141, 1981.

[99] F. Buttussi and L. Chittaro. Locomotion in Place in Virtual Reality: A Comparative Evaluation of Joystick, Teleport, and Leaning. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2019.

[100] Robert Y Wang and Jovan Popović. Real-time hand-tracking with a color glove. *ACM transactions on graphics (TOG)*, 28(3):1–8, 2009.

[101] Charles R Cameron, Louis W DiValentin, Rohini Manaktala, Adam C McElhaney, Christopher H Nostrand, Owen J Quinlan, Lauren N Sharpe, Adam C Slagle, Charles D Wood, Yang Yang Zheng, et al. Hand tracking and visualization in a virtual reality simulation. In *2011 IEEE systems and information engineering design symposium*, pages 127–132. IEEE, 2011.

[102] Victor Adrian Prisacariu and Ian Reid. 3d hand tracking for human computer interaction. *Image and Vision Computing*, 30(3):236 – 250, 2012. Best of Automatic Face and Gesture Recognition 2011.

[103] J. M. Rehg and T. Kanade. Digiteyes: vision-based hand tracking for human-computer interaction. In *Proceedings of 1994 IEEE Workshop on Motion of Non-rigid and Articulated Objects*, pages 16–22, 1994.

[104] V. A. Prisacariu and I. Reid. Robust 3d hand tracking for human computer interaction. In *Face and Gesture 2011*, pages 368–375, 2011.

[105] Giancarlo Iannizzotto, Massimo Villari, and Lorenzo Vita. Hand tracking for human-computer interaction with graylevel visualglove: Turning back to the simple way. In *Proceedings of the 2001 Workshop on Perceptive User Interfaces*, PUI '01, page 1–7, New York, NY, USA, 2001. Association for Computing Machinery.

[106] Eric Krokos, Hsueh-Chien Cheng, Jessica Chang, Bohdan Nebesh, Celeste Lyn Paul, Kirsten Whitley, and Amitabh Varshney. Enhancing Deep Learning with Visual Interactions. *ACM Transactions on Interactive Intelligent Systems (TIIS)*, 1(1), Jul 2018.

[107] Shanxin Yuan, Qi Ye, Bjorn Stenger, Siddhant Jain, and Tae-Kyun Kim. Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4866–4874, 2017.

[108] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Crossing nets: Dual generative models with a shared latent space for hand pose estimation. In *Conference on Computer Vision and Pattern Recognition*, volume 7, 2017.

[109] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4645–4653, 2017.

[110] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images. In *Proceedings of the IEEE international conference on computer vision*, pages 4903–4911, 2017.

[111] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3d hand shape and pose estimation from a single rgb image. In *CVPR*, 2019.

[112] Jonathan Taylor, Richard Stebbing, Varun Ramakrishna, Cem Keskin, Jamie Shotton, Shahram Izadi, Aaron Hertzmann, and Andrew Fitzgibbon. User-specific hand modeling from monocular depth sequences. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 644–651, 2014.

[113] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2540–2548, 2015.

[114] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Trans. Graph.*, 36(6), November 2017.

[115] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: A skinned multi-person linear model. *ACM Trans. Graph.*, 34(6), October 2015.

[116] T. E. de Campos and D. W. Murray. Regression-based hand pose estimation from multiple cameras. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 782–789, 2006.

[117] Srinath Sridhar, Helge Rhodin, Hans-Peter Seidel, Antti Oulasvirta, and Christian Theobalt. Real-time hand tracking using a sum of anisotropic gaussians model. In *Proceedings of the International Conference on 3D Vision (3DV)*, December 2014.

[118] Xiong Zhang, Qiang Li, Hong Mo, Wenbo Zhang, and Wen Zheng. End-to-end hand mesh recovery from a monocular rgb image. In *The International Conference on Computer Vision (ICCV)*, 2019.

[119] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. Pushing the envelope for rgb-based dense 3d hand pose estimation via neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[120] Dominik Kulon, Riza Alp Guler, Iasonas Kokkinos, Michael M. Bronstein, and Stefanos Zafeiriou. Weakly-supervised mesh-convolutional hand reconstruction in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[121] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social motion capture. In *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[122] Christian Zimmermann, Duygu Ceylan, Jimei Yang, Bryan Russel, Max Argus, and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[123] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[124] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), July 2017.

[125] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.

[126] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[127] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[128] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[129] Matt Pharr and Greg Humphreys. *Physically Based rendering, second edition: from theory to implementation.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2010.

[130] Chih-Fan Hsu, Anthony Chen, Cheng-Hsin Hsu, Chun-Ying Huang, Chin-Laung Lei, and Kuan-Ta Chen. Is foveated rendering perceivable in virtual reality?: exploring the efficiency and consistency of quality assessment methods. In *Proceedings of the 2017 ACM on Multimedia Conference*, MM '17, pages 55–63, New York, NY, USA, 2017. ACM.

[131] Matias Koskela, Atro Lotvonen, Markku Mäkitalo, Petrus Kivi, Timo Viitanen, and Pekka Jääskeläinen. Foveated Real-Time Path Tracing in Visual-Polar Space. In Tamy Boubekeur and Pradeep Sen, editors, *Eurographics Symposium on Rendering - DL-only and Industry Track*. The Eurographics Association, 2019.

[132] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. Montage4D: Real-Time Seamless Fusion and Stylization of Multiview Video Textures. *Journal of Computer Graphics Techniques*, 1(15):1–34, Jan. 2019.

[133] Ruofei Du, David Li, and Amitabh Varshney. Geollery: A Mixed Reality Social Media Platform. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, page 13. ACM, May 2019.

[134] Ruofei Du, David Li, and Amitabh Varshney. Project Geollery.com: Reconstructing a Live Mirrored World With Geotagged Social Media. In *Proceedings of the 24th International Conference on Web3D Technology*, Web3D, pages 1–9. ACM, July 2019.

[135] Xuetong Sun, Sarah B. Murthi, Gary Schwartzbauer, and Amitabh Varshney. High-precision 5dof tracking and visualization of catheter placement in evd of the brain using ar. *ACM Trans. Comput. Healthcare*, 1(2), March 2020.

[136] Eric Krokos, Catherine Plaisant, and Amitabh Varshney. Virtual Memory Palaces: Immersion Aids Recall. *Springer VR 2018*, 1(15):1–20, May 2018.

[137] Jonathan Korein and Norman Badler. Temporal Anti-aliasing in Computer Generated Animation. In *Proceedings of the 10th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '83, pages 377–388, New York, NY, USA, 1983. ACM.

[138] M. Sung and S. Choi. Selective Anti-Aliasing for Virtual Reality Based on Saliency Map. In *2017 International Symposium on Ubiquitous Virtual Reality (ISUVR)*, pages 16–19, June 2017.

[139] Chaurasia B.D. and Mathur B.B.L. Eyedness. *Acta Anatomica*, 96(2):301–305, 1976.

[140] D. Y. N. Zotkin, J. Hwang, R. Duraiswaini, and L. S. Davis. HRTF personalization using anthropometric measurements. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*, pages 157–160, Oct 2003.

[141] Hiroshi Ono and Raphael Barbeito. The cyclopean eye vs. the sighting-dominant eye as the center of visual direction. *Perception & Psychophysics*, 32(3):201–210, 1982.

[142] Tomer Elbaum, Michael Wagner, and Assaf Botzer. Cyclopean vs. dominant eye in gaze-interface-tracking. *Journal of Eye Movement Research*, 10(1), Jan. 2017.

[143] Zhenping Xia and Eli Peli. 30-1: Cyclopean eye based binocular orientation in virtual reality. *SID Symposium Digest of Technical Papers*, 49(1):381–384, 2018.

[144] Franziska Mueller, Micah Davis, Florian Bernard, Oleksandr Sotnychenko, Mickeal Verschoor, Miguel A. Otaduy, Dan Casas, and Christian Theobalt. Real-time Pose and Shape Reconstruction of Two Interacting Hands With a Single Depth Camera. *ACM Transactions on Graphics (TOG)*, 38(4), 2019.